UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INFORMÁTICOS

CAMPUS MONTEGANCEDO

TRABAJO DE FIN DE MÁSTER

MÁSTER UNIVERSITARIO EN SOFTWARE Y SISTEMAS

CIFRADO DE IMÁGENES DIGITALES BASADO EN TEORÍA DEL CAOS – MAPAS LOGÍSTICOS

POSTULANTE: Marcos Espinoza Illanes

TUTOR: Raúl Alonso

MADRID, JULIO 2015

DEDICATORIA

A Dios,
Mi Salvador y guía en mi proyecto de vida que ha iluminado toda acción, palabra y pensamiento; siendo Él obrando a través de mí.

AGRADECIMIENTOS

Mis más sinceros agradecimientos:

A mi Padre de los Cielos por guiarme y acompañarme durante toda la vida; y llenarme de sabiduría y fortaleza durante este proyecto.

A mi papá Ramiro Espinoza Delgado por enseñarme con palabras y acciones que todo esfuerzo siempre tiene su recompensa; convirtiéndose en mi referente y modelo de hombre a seguir.

A mi mamá María Elena Illanes de Espinoza por alentarme en los momentos de mayor debilidad y aconsejarme en los momentos de mayor incertidumbre.

A mi hermanita Andy porque hizo mis preocupaciones suyas y su apoyo incondicional fueron otro incentivo para seguir adelante.

Al Ing. Raúl Alonso, por haber creído en mis capacidades al inicio; por haberme apoyado académicamente y disuelto dudas durante el proceso; y por haberme acompañado en la satisfacción de haber culminado nuestro proyecto.

Finalmente, pero no menos importante, a mis amigos de curso de Máster por compartir todos los estos meses llenos de anécdotas inolvidables y por su constante apoyo moral y académico.

RESUMEN

Las nuevas tendencias de compartir archivos multimedia a través de redes abiertas, demanda el uso de mejores técnicas de encriptación que garanticen la integridad, disponibilidad y confidencialidad, manteniendo y/o mejorando la eficiencia del proceso de cifrado sobre estos archivos. Hoy en día es frecuente la transferencia de imágenes a través de medios tecnológicos, siendo necesario la actualización de las técnicas de encriptación existentes y mejor aún, la búsqueda de nuevas alternativas.

Actualmente los algoritmos criptográficos clásicos son altamente conocidos en medio de la sociedad informática lo que provoca mayor vulnerabilidad, sin contar los altos tiempos de procesamiento al momento de ser utilizados, elevando la probabilidad de ser descifrados y minimizando la disponibilidad inmediata de los recursos.

Para disminuir estas probabilidades, el uso de la teoría de caos surge como una buena opción para ser aplicada en un algoritmo que tome partida del comportamiento caótico de los sistemas dinámicos, y aproveche las propiedades de los mapas logísticos para elevar el nivel de robustez en el cifrado.

Es por eso que este trabajo propone la creación de un sistema criptográfico basado sobre una arquitectura dividida en dos etapas de confusión y difusión. Cada una de ellas utiliza una ecuación logística para generar números pseudoaleatorios que permitan desordenar la posición del píxel y cambiar su intensidad en la escala de grises. Este proceso iterativo es determinado por la cantidad total de píxeles de una imagen. Finalmente, toda la lógica de cifrado es ejecutada sobre la tecnología CUDA que permite el procesamiento en paralelo.

Como aporte sustancial, se propone una nueva técnica de encriptación vanguardista de alta sensibilidad ante ruidos externos manteniendo no solo la confidencialidad de la imagen, sino también la disponibilidad y la eficiencia en los tiempos de proceso.

ABSTRACT

New trends to share multimedia files over open networks, demand the best use of encryption techniques to ensure the integrity, availability and confidentiality, keeping and/or improving the efficiency of the encryption process on these files. Today it is common to transfer pictures through technological networks, thus, it is necessary to update existing techniques encryption, and even better, the searching of new alternatives.

Nowadays, classic cryptographic algorithms are highly known in the midst of the information society which not only causes greater vulnerability, but high processing times when this algorithms are used. It raise the probability of being deciphered and minimizes the immediate availability of resources.

To reduce these odds, the use of chaos theory emerged as a good option to be applied on an algorithm that takes advantage of chaotic behavior of dynamic systems, and take logistic maps' properties to raise the level of robustness in the encryption.

That is why this paper proposes the creation of a cryptographic system based on an architecture divided into two stages: confusion and diffusion. Each stage uses a logistic equation to generate pseudorandom numbers that allow mess pixel position and change their intensity in grayscale. This iterative process is determined by the total number of pixels of an image. Finally, the entire encryption logic is executed on the CUDA technology that enables parallel processing.

As a substantial contribution, it propose a new encryption technique with high sensitivity on external noise not only keeping the confidentiality of the image, but also the availability and efficiency in processing times.

CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

CAPÍTULO I

GENERALIDADES

1.1. INTRODUCCIÓN

El conocimiento humano ha evolucionado a lo largo de los años a consecuencia de la reproducción y proliferación de la información. En otras palabras, después de generar la información, es inminente la propagación de la misma a otros entes para su manipulación o consumo final. Una información puede estar representada de distintas maneras; desde una carta manuscrita, sonido e incluso a través de una imagen porque representa un mensaje que contiene significado entendible para el humano. La información puede ser heterogénea, por ejemplo, reportes contables bancarios, informes gubernamentales de inteligencia, retratos hablados policiales, radiografías o informes médicos. De esta manera, se deduce que es el contexto y el uso de la información lo que define el nivel de criticidad al compartir el mensaje. Los recursos que portan información sensible adoptan riesgos a lo largo de su ciclo de vida como por ejemplo la alteración, distorsión e incluso el mal uso de los mismos Por lo tanto, divulgar información sensible requiere ser protegida para minimizar dichos riesgos.

La digitalización de imágenes no solo supone un avance de la tecnología, sino también nuevas responsabilidades sobre su aplicación. Desde que Russell Kirsch creó una imagen digital de 176×176 pixeles en 1957 (National Institute of Standards and Technology, 2009), fue imperiosa e inevitable la necesidad de compartir este tipo de recursos a través de medios electrónicos y de esa manera cumplir con el ciclo de vida de la información. Actualmente, el alto crecimiento en las tecnologías de comunicación, más específicamente Internet, ocasiona que el intercambio de recursos digitales incremente drásticamente y de esta manera la información sea más vulnerable (Sharma & Kowar, 2010). Aplicaciones como la televisión de pago, video conferencias o información médica/militar clasificada son motivos suficientes para proteger este tipo de información mientras son transmitidos. Por lo tanto, la investigación en la propagación de imágenes digitales supone entender la importancia de la confidencialidad y preservar la privacidad del usuario.

Figura 1: Primera imagen digital de 176 x 176 pixeles

Fuente: (National Institute of Standards and Technology, 2009)

Denegar el acceso desautorizado a la información es tarea de seguridad, la misma se mide bajo los parámetros de integridad, confidencialidad y disponibilidad sobre el dato (Ephin, Joy, & Vasanthi, 2013). Bajo integridad se entiende la no alteración de la información por ningún intruso. Respecto a la confidencialidad, advierte que el acceso a la información si y solo si puede lograrse por la(s) persona(s) autorizadas. Finalmente la disponibilidad supone que la seguridad sobre un recurso compartido no debe afectar el acceso a la misma cuando así se lo requiera. Por lo tanto, mientras se cumplan los tres principios básicos de seguridad sobre los recursos compartidos como las imágenes, se puede garantizar cierto nivel de mitigación sobre los riesgos inherentes en redes abiertas.

La encriptación de información surge como una herramienta de seguridad para garantizar que los recursos que transitan en redes abiertas, especialmente Internet queden protegidas de terceros (Sankpal & Vijaya, 2014). Representa un proceso en el cual se aplica algoritmos matemáticos especiales y “keys”, o claves, para transformar la información original en un código cifrado. De la misma manera, la desencriptación supone el proceso a la inversa para obtener la información

original. La aplicación de esta herramienta es muy común para promover la seguridad en imágenes ya que convierte una imagen original en otra imagen difícil de entender, incluso de deducir. Bajo este entendido, la encriptación hace uso de la criptografía para cifrar y descifrar información de manera controlada y segura, garantizando mantener la privacidad de un recurso que circula por medios digitales.

Criptografía es la ciencia que protege la privacidad de la información durante la transmisión bajo condiciones hostiles (Philip & Das, Survey: Image Encryption using Chaotic Cryptography Schemes, 2011). Cifrar una imagen es distinto a cifrar un texto plano. Existen algoritmos, técnicas y/o cifradores convencionales como AES[1], DES[2], RSA[3], IDEA[4], LFSR[5], etc que consideran un texto en plano como un bloque cifrado o un "stream" de datos (Delfs & Knebl, 2006); siendo que una imagen o video contiene un volumen grande de datos y fuerte correlación entre los pixeles y ésta es la razón por la cual no son adecuados para encriptar imagen o video en tiempo real. De esta manera, la encriptación basada en caos sugiere una nueva tendencia para afrontar el cifrado de una imagen bajo los indicadores de niveles de seguridad y tiempos de procesamiento.

A lo largo de las últimas décadas, investigadores dieron cuenta de la existencia de una fuerte relación entre el caos y la criptografía (Pisarchik & Zanin, 2008). En sistemas reales, el caos y el ruido son dos comportamientos naturales e irregulares cuyas principales propiedades son alta dependencia en las condiciones iniciales y parámetros del sistema, naturaleza pseudo-aleatoria, sin periodicidad, entre otras. Muchas de estas características se acoplan perfectamente a los requerimientos de la criptografía, como lo son la difusión y confusión. Desde este punto de vista, la encriptación a través del caos es una alternativa que logra cumplir con el objetivo de proteger una imagen.

[1] Advanced Encryption Standard
[2] Data Encryption Standard
[3] Rivest, Shamir y Adleman
[4] International Data Encryption Algorithm
[5] Linear Feedback Shift Register

Con la finalidad de enlazar la encriptación con las propiedades de la teoría del caos, se hace uso de los mapas caóticos expresados a través de funciones. El mapa logístico es uno de los modelos discretos dinámicos más simples expresado en la siguiente función (Jiménez Rodríguez, Jaimes Reategui, & N. Pisarchik, 2012):

$$x_{n+1} = ax_n(1 - x_n)$$

Sin embargo, según (Ephin, Joy, & Vasanthi, 2013), una de las desventajas en el uso de los mapas logísticos se relaciona con el tiempo de proceso. Para afrontar este alcance, en el presente trabajo se propone la utilización de una plataforma de computación paralela denominada CUDA[6] que incrementa dramáticamente el desempeño a través de una nueva unidad de procesamiento de imágenes como lo es GPU[7] (NVIDIA Corporation, 2015). De esta manera, se provee de una solución que mantiene las propiedades deseadas por la criptografía y mejora tiempos de proceso para cumplir con la disponibilidad que exige la seguridad.

1.2. MOTIVACIÓN

Vivimos en un tiempo donde el acceso a la tecnología invade la privacidad de las personas que hacen uso de esta herramienta. Un alto porcentaje de los recursos multimedia son imágenes que son utilizados en muchas áreas como autenticación biométrica, ciencias médicas, militares e incluso fotografías personales. Este tipo de recursos cuando viajan sin ningún tipo de seguridad por redes abiertas se exponen a un alto riesgo de ser no solo interceptadas por terceros, sino también manipuladas y alteradas parcial o totalmente. Lo que es más, mientras las resoluciones de las imágenes digitales aumenten la masa de información en los pixeles, el proceso de encriptación y/o cifrado de las mismas supondrá mayores tiempos de procesamiento por esta razón lógica. Por lo tanto, es imperiosa la necesidad de iniciar investigaciones sobre nuevos algoritmos de encriptación más robustos y de menos tiempo de proceso. Con estas propiedades, se incrementa la

[6] Compute Unified Device Architecture
[7] Graphics Processing Unit

dificultad para terceros en su intención de alterar la información a través de la ejecución de algoritmos agiles en tiempo. En conclusión, la razón que motiva a realizar este aporte es la importancia de proteger las imágenes del acceso desautorizado mejorando niveles de cifrado sin desmedro del tiempo de ejecución.

1.3. PLANTEAMIENTO DEL PROBLEMA

El planteamiento del problema gira en torno a un problema central, al cual se adhieren varios problemas secundarios que lo complementan.

1.3.1. Problema principal

La creciente tendencia a compartir imágenes y el riesgo que supone utilizar redes abiertas como Internet, provoca que los algoritmos de encriptación tradicionales ya no sean adecuados[8] para garantizar la privacidad del contenido gráfico que transita por canales inseguros.

1.3.2. Problemas secundarios

- Las nuevas herramientas tecnológicas sumadas al ingenio de hackers y afines facilita el descifrado de mensajes encriptados con algoritmos tradicionales.
- Las distintas dimensiones y calidad de imágenes que actualmente son generadas desde varios dispositivos, consume mayor tiempo en el proceso de encriptación debido a la cantidad de información que contienen.
- La técnica de ataque prueba-error y su probabilidad de acertar el valor de las llaves secretas, eleva el riesgo de descifrar parcialmente el mensaje.

1.4. OBJETIVOS

Al igual que el planteamiento del problema, el objetivo se concentra en un objetivo principal y varios objetivos específicos. Uno para cada problema secundario.

[8] Adecuados en términos de tiempo y seguridad. La encriptación es adecuada cuando es efectiva y robusta.

1.4.1. Objetivo principal

Desarrollar una solución criptográfica eficiente y robusta para minimizar los riesgos de seguridad sin desmedro del tiempo de ejecución garantizando disponibilidad del recurso y confidencialidad en el contenido circulante en redes inseguras.

1.4.2. Objetivos específicos

- Construir un algoritmo de cifrado cuyo proceso de encriptación aplique conceptos de confusión y difusión para proporcionar una fuerte seguridad criptográfica.
- Mejorar los tiempos de ejecución no sólo en el proceso de encriptación, sino también en el proceso de desencriptación mediante el uso de tecnologías de procesamiento paralelo de imágenes como CUDA[9] para no afectar la disponibilidad del recurso.
- Elevar la sensibilidad en las condiciones iniciales ante ruidos externos para que una aproximación en los parámetros por parte del hacker no cause una imagen parcialmente descifrada.

1.5. ORGANIZACIÓN DE LA MEMORIA

El presente capitulo incluye la introducción y la motivación. Posteriormente se ha identificado el problema principal y secundarios; y de la misma manera los objetivos acordes a los anteriores.

Capítulo 2 contiene el estado del arte en la que se describen varios antecedentes relacionados con la temática principal del presente trabajo. Concretamente se realiza la revisión de once artículos más relevantes que influyeron en las conclusiones del trabajo de investigación. Al finalizar este capítulo, se presenta una tabla de resumen que relaciona todos los artículos e identifica tanto sus ventajas como desventajas.

[9] Compute Unified Device Architecture

Para el capítulo 3, se proporciona la información como fundamento en el cual está basada el trabajo. Contiene conceptos básicos de las imágenes digitales y principalmente sobre la teoría del caos, sus propiedades y los diferentes sistemas caóticos. De igual manera, se describe en detalle los mapas logísticos que son utilizados como parte fundamental del algoritmo. Finalmente, la investigación bibliográfica hace referencia a la criptografía, sus características y la estrecha relación que tiene con el caos.

En el capítulo 4 se detalla las tecnologías utilizadas para el desarrollo y ejecución del algoritmo de encriptación. Específicamente, se describe la utilidad de OpenCV dentro del trabajo y las funciones expuestas por las librerías utilizadas. De igual manera, se presenta las denominadas GPU's como dispositivo de computación de datos en paralelo resaltando la arquitectura sobre la cual se ejecuta. La arquitectura utilizada para el cálculo de GPU es CUDA y este capítulo contiene una descripción del procesamiento y la implementación.

Capítulo 5 contiene la explicación del algoritmo en su totalidad. Todas las consideraciones y procedimientos incluidos en el proceso de cifrado son descritas en esta sección. La primera parte describe de manera general la arquitectura sobre la cual se desarrolla la lógica del sistema criptográfico. A continuación los algoritmos tanto de encriptación como desencriptación se presentan a detalle. Para finalizar el capítulo, se expone la innovación realizada en este nuevo algoritmo desarrollado y las influencias por parte de otros autores que fueron tomadas en cuenta para la construcción de la solución.

El capítulo 6 contiene los resultados obtenidos de la ejecución de la solución en base a imágenes de distintos tamaños. En primer lugar se muestran las pruebas de funcionamiento básico de la solución para verificar el correcto procesamiento. En segundo lugar, las pruebas se enfocan en demostrar la generación del caos y la manera en la que afectan a los resultados de la encriptación. La última parte de las pruebas trata del cálculo y comparación del tiempo de ejecución en la que se utilizan imágenes de distintos tamaños y estándares; la comparación de tiempos se realiza entre procesamiento en CPU y procesamiento en GPU

Finalmente, en el capítulo 7 se establece las conclusiones del trabajo enmarcados en los objetivos propuestos. De igual manera se sugiere posibles líneas de trabajo con la finalidad de continuar la investigación sobre el mismo tema en un futuro.

CAPÍTULO II

ESTADO DEL ARTE

2.1. ANTECEDENTES

Las técnicas para la encriptación de imágenes fueron creadas hace muchos años y utilizado en muchos proyectos. Muchos de ellos han evolucionado conforme ha pasado el tiempo y por tanto se han generado distintas soluciones a una misma problemática de seguridad. En su mayoría, se ha identificado ciertas ventajas y desventajas que son tomados a consideración para la elaboración de la solución final. Por lo tanto, a continuación se describirán los antecedentes que darán el punto de partida a la investigación:

- ***Antecedente 1:***

Los autores de este artículo proponen un nuevo método para la encriptación de imágenes que resulta de la combinación de FRPT[10] y WPT[11]. El método inicia mediante la descomposición de la imagen inicial en varias sub-bandas. Posteriormente, se selecciona de forma aleatoria y son encriptadas mediante FRPT, es decir, mediante el uso del Fraccional de Fourier. Esta técnica es más efectiva que WPT porque la información es aleatoriamente mejor distribuida. La ventaja de esta técnica radica en lograr alta confidencialidad de la información. Una de las desventajas es el espacio limitado para las llaves (Keys). Un esquema de encriptación adecuado debiera contemplar una llave suficientemente larga como para combatir a un ataque de fuerza bruta. La otra desventaja es la calidad perceptual limitada en el proceso de recuperación de la imagen original. (Chen & Zhao, 2008)

- ***Antecedente 2:***

Basados en el inconveniente de una débil seguridad que proporcionaban lo sistemas criptográficos caóticos de una sola dimensión, los autores del artículo (Gao, Zhang, Liang, & Li, 2006) presentaron un nuevo algoritmo denominado NCA[12]. A diferencia del mapa caótico de una sola dimensión que utiliza una

[10] Fractional Wavelet Packet Transform
[11] Wavelet Packet Transform
[12] Nonlinear Chaotic Algorithm

función lineal, NCA utiliza funciones tangenciales y exponenciales. El proceso de encriptación inicia estableciendo las llaves y posteriormente itera 100 veces para obtener la imagen encriptada, siendo un valor fijo y no parametrizable. Al finalizar el proceso, la imagen es enviada por un canal público de comunicación y las llaves de encriptación a través de un canal de comunicación seguro. Las ventajas de este artículo son el alto nivel de seguridad y la sensibilidad de las llaves.

- ***Antecedente 3:***

La idea básica que pretenden mostrar los autores en el artículo (Sun, Liu, Li, & Lu, 2008) es encriptar la imagen basada en un mapa caótico pixel a pixel y después los pixeles son cambiados de posición en múltiples direcciones del espacio. Inicialmente, la imagen original es trasformada en una matriz y el resultado es encriptado usando los resultados de la iteración anterior. Utilizando las condiciones iniciales y los parámetros del mapa caótico, el mapa de caos espacial es iterado por primera vez y una nueva matriz es generada, en el siguiente paso las condiciones iniciales son modificadas. Este proceso es repetido por cierto tiempo y finalmente se obtiene la imagen cifrada. El proceso de desencriptación es similar a la de encriptación utilizando la imagen cifrada como entrada en lugar de la imagen original en el proceso de encriptación. Debido a que ambos procesos (Encriptación y Desencriptación) tienen la misma estructura, la complejidad y el tiempo que consumen es el mismo. Sus características principales es la alta seguridad que proporciona y el tamaño de las llaves suficientemente larga para resistir ataques de fuerza bruta.

- ***Antecedente 4:***

El esquema de encriptación de imagen presentado en este artículo, utiliza una llave secreta externa de 80 bits y dos mapas logísticos. Las condiciones iniciales de ambos mapas logísticos son obtenidas en base a la llave secreta externa. En el algoritmo, el primer mapa logístico es utilizado para generar números entre el rango del 1 al 24 (los números puedes ser repetidos). Haciendo uso de los números generados por el primer mapa logístico, las condiciones iniciales del

segundo mapa logístico son modificadas. Esta lógica permite que el algoritmo adquiera mayor aleatoriedad de forma dinámica. Por lo tanto, para encriptar los pixeles, el proceso propuesto utiliza 8 diferentes tipos de operaciones de los cuales uno de ellos será utilizado para un pixel en particular el cual es seleccionado a través del mapa logístico. Después de encriptar cada bloque de 16 pixeles, la llave secreta es modificada para que el cifrador sea más robusto contra cualquier ataque. Sin duda alguna, este algoritmo tiene alta sensibilidad a cambios en la llave secreta, aun con una capacidad pequeña para el mismo. El tamaño de llave puede ascender hasta $2^{80} (\approx 1.20893 \times 10^{24})$. (Pareek, Patidar, & Sud, 2006)

- ***Antecedente 5:***

Este artículo hace uso del llamado "Chaotic Cat Map" de dos dimensiones que es generalizado a un esquema de encriptación 3D para el diseño de un esquema de encriptación simétrica en tiempo real. En un inicio se selecciona una secuencia de 128 bits como llave y son separados en 8 grupos los cuales son mapeados para los diversos parámetros del "3D cat map" y el mapa logístico. A continuación, la imagen de dos dimensiones a cambiada a una imagen de tres dimensiones. Después, utilizando el mapa de tres dimensiones, se genera imágenes barajadas. Posteriormente, se utiliza el mapa logístico para ejecutar una operación de suma XOR sobre las imágenes obtenidas y de esta manera transformar los cubos de tres dimensiones a la imagen de dos dimensiones como resultado de la encriptación. En comparación con otras técnicas, esta técnica tiene mayor velocidad de encriptación pero al mismo tiempo menor tamaño de llave. Su tamaño máximo asciende a $2^{128} (\approx 3.4028 \times 10^{38})$. (Chen, Mao, & Chui, 2004)

- ***Antecedente 6:***

El artículo presentado por Yaobin Mao, Guanrong Chen y Shiguo Lian tiene alguna similitud con el anterior debido al uso de un mapa caótico bidimensional que es generalizado a uno en 3D. Al inicio se selecciona una secuencia de 128 bits como la llave, y los agrupa en seis grupos los cuales son mapeados en varios parámetros. Después la imagen de dos dimensiones es cambiada a una imagen

3D. A continuación, el utiliza el mapa tridimensional denominado “Baker map[13]” el cual generará imágenes barajadas. Posteriormente, el mapa logístico ejecuta la operación de sumatoria XOR cada imagen barajada. Finalmente, para obtener la imagen encriptada, se trasforma los cubos de tres dimensiones en una imagen de dos dimensiones. El tamaño máximo de la llave asciende a $2^{128}(\approx 3.4028 \times 10^{38})$. (Mao, Chen, & Lian, 2004)

- ***Antecedente 7:***

Los autores de este artículo presentan un algoritmo de encriptación basado en la trasformada de Fourier y sistema caótico. El proceso de encriptación consta de dos pasos: el primero de ellos encripta la imagen por duplicado de manera aleatoria usando el dominio fraccional de Fourier. El segundo paso consiste en encriptar nuevamente utilizando la matriz generada por el sistema caótico y de esa manera se obtiene la imagen de las huellas dactilares encriptada. El proceso de desencriptación es similar a la encriptación de forma inversa. Aun con el tamaño de la llave pequeña (10^{60}), soporta ataques de fuerza bruta. (Lai, Liang, & Cui, 2010)

- ***Antecedente 8:***

Otro esquema para encriptar imágenes de huellas dactilares eficiente y seguro es presentado en este artículo. Los autores proponen una combinación de operaciones para barajear pixeles y un sistema dinámico no-lineal de caos. Al inicio, una imagen totalmente barajada es utilizada para intercambiar las posiciones de todos sus pixeles en un dominio espacial. Posteriormente, los pixeles de la imagen resultante son organizados ordenadamente de izquierda a derecha y de arriba abajo. A continuación, es convertido a una representación binaria. Después, se hace uno de un filtrador caótico llamado NDF[14] para ser iterado. Para obtener la imagen encriptada, las llaves son generadas por el NDF y la operación XOR en conjunto con la representación binaria de la imagen. El

[13] Lleva el nombre de una operación de amasado que los panaderos se aplican a la masa
[14] Nonlinear Digital Filter

algoritmo de desencriptación es similar a la encriptación, es decir, se utiliza NDF con los mismos parámetros y valores iniciales utilizados en la encriptación. Finalmente, reordenar los pixeles a su posición original para obtener la imagen original. Aunque el algoritmo resiste ataques de fuerza bruta y es sensible a las llaves, no es eficiente en términos de tiempo. (Zhao, Li, & Yan, 2008)

- ***Antecedente 9:***

En este artículo se desarrolló un algoritmo de encriptación de imágenes basada en un plano de bits. El algoritmo consiste en dos operaciones: la distorsión de la imagen y la encriptación LSB[15]. Para la primera operación, se distorsiona la imagen completa de la huella dactilar mediante el uso de operaciones muy simples. Para cada pixel, se escoge su LSB como ruido aleatorio y se genera el plano de bits LSB. A continuación, se utiliza un XOR exclusivo del plano de bits y todos lo pixeles de la imagen de las huellas dactilares. El plano de bits LSB es esencial ya que sin ella, el atacante no puede recuperar la estructura de la imagen a partir de la imagen distorsionada. Por lo tanto, en la encriptación LSB, solo es necesario encriptar el plano de bits LSB utilizando llaves compartidas. Dado que el cliente y el sensor comparten la misma llave, el cliente puede recuperar la imagen original de huellas dactilares a través de la desencriptación del plano LSB y después aplicando la misma operación XOR exclusiva. Se garantiza total confidencialidad de la imagen impresa de las huellas dactilares entre el sensor y el cliente en tiempo real. La desventaja es la calidad perceptual del resultado. (Moon, Chung, Pan, Moon, & Chung, 2006)

- ***Antecedente 10:***

Los autores de este artículo proponen el uso de una matriz totalmente barajada para intercambiar las posiciones de los pixeles de la imagen original y posteriormente usar una combinación de estados provenientes de dos sistemas caóticos para confundir la relación entre la imagen en plano y la imagen cifrada. La imagen convertida en la matriz barajada es generada mediante varias iteraciones

[15] Least Significant Bit

del mapa logístico y el intercambio de posiciones entre filas y columnas de la matriz. Finalmente la imagen cifrada es obtenida utilizando el sistema caótico de Lorenz y el sistema caótico de Chen. Este algoritmo de encriptación tiene espacio amplio para las llaves lo que lo convierte en sensible a la misma y resisten a ataques de fuerza bruta. Sin embargo el tiempo de encriptación no es muy eficiente debido a que la transformación de fila a columna requiere mucho tiempo. (Gao & Chen, 2008)

- ***Antecedente 11:***

El autor Delong Cui presenta un algoritmo de encriptación que consiste en dos pasos. En un inicio, la imagen es encriptada en dos fases aleatorias utilizando el dominio fraccional de Fourier. Posteriormente, la misma imagen es encriptada utilizando una matriz generada por el mapa logístico y finalmente se obtiene la imagen de huellas dactilares encriptada. Es resistente a ataques de fuerza bruta pero la longitud de la llave es pequeña (10^{60}). (Cui, 2010)

A continuación una tabla resumen de los antecedentes descritos y sus principales atributos analizados

Tabla 1: Resumen de antecedentes

N°	Autores, año publicación	Método	Ventaja	Desventaja
1	(Chen & Zhao, 2008)	• Fractional Wavelet Packet Transform	• Alta confiabilidad en la información	• Espacio limitado en la clave • Calidad perceptual limitada
2	(Gao, Zhang, Liang, & Li, 2006)	• Algoritmo caótico no lineal	• Alto nivel de seguridad	• Tamaño de clave pequeño
3	(Sun, Liu, Li, & Lu, 2008)	• Mapa caótico espacial	• Alto nivel de seguridad	• Tamaño de clave pequeño
4	(Pareek, Patidar, & Sud, 2006)	• Dos mapas caóticos logísticos	• Alta sensibilidad en las claves	• Tamaño de clave pequeño
5	(Chen, Mao, & Chui, 2004)	• Mapa caótico “Cat” (3D) • Mapa logístico	• Alta velocidad en la encriptación	• Tamaño de clave pequeño • La imagen original debe ser de tamaño cuadrado
6	(Mao, Chen, & Lian, 2004)	• Mapa caótico “Baker” (3D) • Mapa logístico	• Alta velocidad en la encriptación	• Tamaño de clave pequeño • La imagen original debe

				ser de tamaño cuadrado
7	(Lai, Liang, & Cui, 2010)	• Transformada Fraccional de Fourier • Mapa logístico	• Alta resistencia a ataques de fuerza bruta	• Tamaño de clave pequeño
8	(Zhao, Li, & Yan, 2008)	• Mapa logístico • Mapa caótico no lineal	• Alta resistencia a ataques de fuerza bruta • Algoritmo de encriptación es sensible a las claves	• Tiempo de proceso ineficiente
9	(Moon, Chung, Pan, Moon, & Chung, 2006)	• Encriptación en plano de bits	• Garantiza la confidencialidad	• Calidad perceptual limitada
10	(Gao & Chen, 2008)	• Mapa logístico • Sistema caótico de Lorenz	• Tamaño grande de clave • Sensible en la clave • Resiste ataques de fuerza bruta	• Tiempo de proceso ineficiente
11	(Cui, 2010)	• Transformada Fraccional de Fourier • Mapa logístico	• Resiste ataques de fuerza bruta	• Tamaño de clave pequeño

Fuente: (Ephin, Joy, & Vasanthi, 2013)

CAPÍTULO III

MARCO TEÓRICO

3.1. FUNDAMENTOS DE IMAGEN DIGITAL

Dentro del entorno de los recursos multimedia que circulan por redes de comunicación digital, el presente trabajo se enfoca en las imágenes y más propiamente dichas imágenes digitales. Por esta razón se detallan tanto su definición como características del mismo.

3.1.1. Imagen Analógica

Antes de generar imágenes digitales, se debe tener consideración de los procesos técnicos que comprenden convertir una representación analógica en digital. De igual manera se debe tomar en cuenta los atributos del documento fuente como por ejemplo las dimensiones físicas y presentación, nivel de detalles, rango tonal y presencia de color. Otra característica de los documentos es el proceso de producción para crearlos que pueden ser medios manuales, mecánicos, fotográficos y, ahora en auge, electrónicos. En otras palabras, una imagen natural capturada con una cámara, un telescopio, un microscopio o cualquier otro tipo de instrumento óptico presenta una variación de sombras y tono continuo. Estas imágenes son denominadas imágenes analógicas. (Biblioteca de la Universidad de Cornell / Departamento de Investigación, 2003)

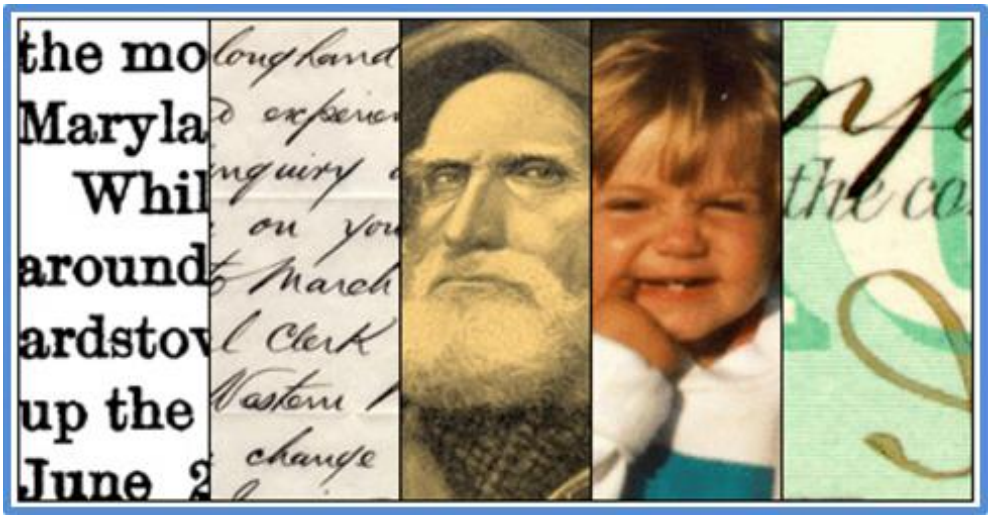
Las imágenes analógicas con formato de papel y película se categorizan en cinco grupos que afectan su transformación digital, los mismos son:

- Texto impreso / dibujos de líneas simples: se refiere a la representación en base a bordes definidos, sin variación de tono. Por ejemplo un libro que contiene textos y gráficos de líneas simples.
- Manuscritos: se refiere a la representación en base a bordes suaves que producen a mano o a máquina, sin embargo no se definen los bordes típicos. Por ejemplo el dibujo de una letra o una línea.
- Media tinta: se refiere a los materiales gráficos o fotográficos representados por una cuadrícula con un esquema de puntos o líneas de diferente tamaño y espaciadas regularmente que, habitualmente se encuentran en un ángulo.

También incluye algunos tipos de arte gráfica, como por ejemplo, los grabados.

- Tono Continuo: se refiere a los elementos tales como fotografías, acuarelas y algunos dibujos de líneas finamente grabadas que exhiben tonos que varían suave o sutilmente.
- Combinado: se refiere a los documentos que contienen dos o más de las categorías mencionadas anteriormente, como por ejemplo, los libros ilustrados.

Figura 2: Tipos de documentos, de izquierda a derecha. Texto impreso, manuscrito, media tinta, tono continuo y combinado



Fuente: (Biblioteca de la Universidad de Cornell / Departamento de Investigación, 2003)

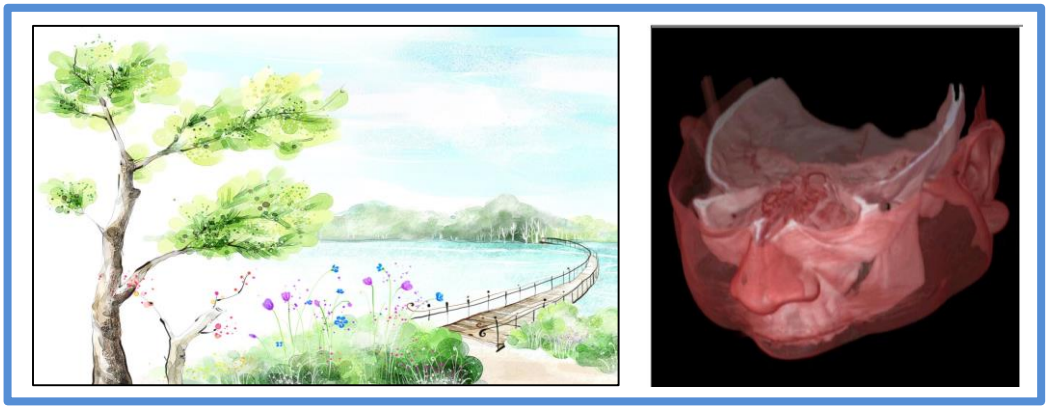
3.1.2. Imagen Digital

Para que una imagen analógica, con sus respectivos atributos descritos en el punto anterior, pueda ser manipulada a través de un ordenador, primero debe ser trasformado al formato adecuado. Este formato es la imagen digital. La trasformación de una imagen analógica a una discreta se denomina digitalización y corresponde el primer paso antes de procesar imágenes. Por tanto, una imagen digital se define como la representación gráfica basada en bits (0 y 1) que pueden ser procesadas por los ordenadores ya que manipula información numérica. (Biblioteca de la Universidad de Cornell / Departamento de Investigación, 2003)

Las imágenes digitales se clasifican tomando en cuenta dos aspectos: la dimensión y la paleta de colores. Considerado el primer aspecto, las imágenes puedes ser:

- Dos dimensiones: son aquellas imágenes desde las cuales no solo se pueden generar vectores, sino mapa de bits que es un concepto a utilizar en el presente trabajo.
- Tres dimensiones. Son aquellas imágenes que se representan en el espacio. Comúnmente utilizadas en las áreas médicas. Este tipo de imágenes no se encuentra dentro el alcance del trabajo.

Figura 3: Dimensiones de imágenes, de izquierda a derecha. Dos Dimensiones y Tres Dimensiones



Fuente: (2D) http://wallpoper.com/ - **(3D)** http://www.3d-doctor.com

Tomando en consideración el aspecto de los colores, las imágenes digitales se clasifican en:

- Imágenes binarias: Son aquellas imágenes cuyos valores de pixel, únicamente pueden ser 0 o 1.
- Imágenes en escala de grises: Son aquellas imágenes cuyos valores representan el valor tonal y puede oscilar entre 0 y 255. Siendo 0 el valor negro y 255 el blanco.

- Imágenes a color: cada pixel de este tipo de imágenes está compuesto por tres vectores que representan los colores básicos RGB[16]. Cada elemento del vector oscila entre 0 y 255 y la combinación de los tres vectores produce un color específico para un pixel de la imagen.

Figura 4: Tipos de imágenes, de izquierda a derecha. Imagen binaria, Imagen en escala de grises e Imagen a color

Fuente: http://www.unioviedo.es/

3.1.3. Pixeles

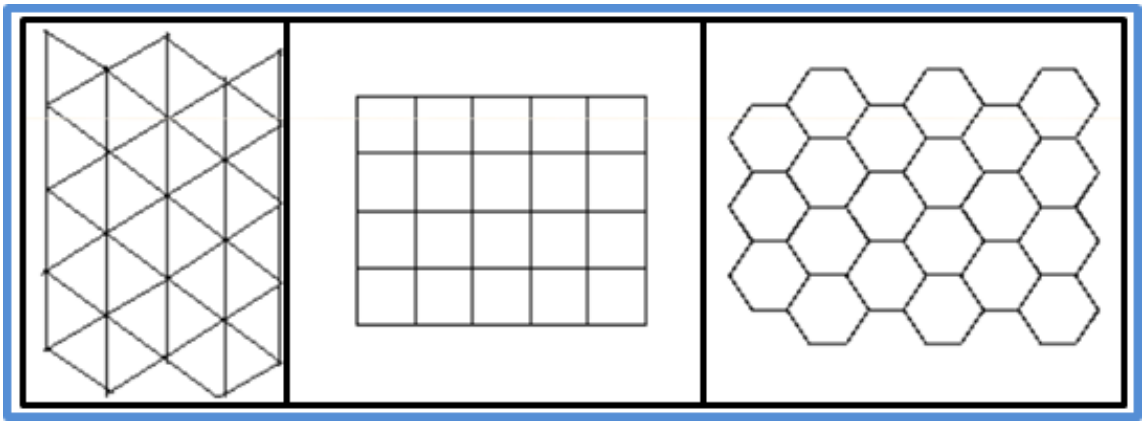
El denominativo “pixel” proviene de la unión de las palabras en inglés “picture” y “element”. Son consideradas como las unidades de color que componen una imagen digital. Un pixel no lleva una medida concreta como otras unidades de medida como por ejemplo 1mm, o 1cm, simplemente es la medida de división de una retícula en celdillas que a continuación se explica.

Dentro el proceso de digitalización, existe una operación denominada muestreo que consiste en una subdivisión de la imagen analógica en porciones. Debido a que el trabajo se enfoca en imágenes 2D, las formas geográficas que adoptan la subdivisión pueden ser: triángulos, cuadrados y hexágonos. Los polígonos conformados representan sensores sensibles a la intensidad de la luz. De esta manera, las imágenes digitales se generan a partir de una retícula rectangular

[16] Red, Green y Blue

formada por cedillas. (Gobierno de España - Ministerio de Educación,Cultura y Deporte, 2015)

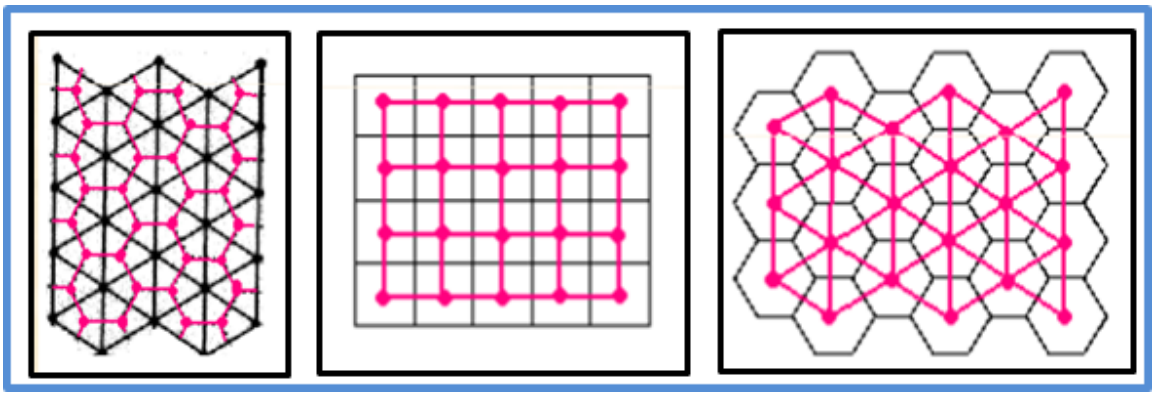
Figura 5: Tipos de mallado, de izquierda a derecha. Mallado triangular, Mallado cuadrangular y Mallado hexagonal



Fuente: Elaboración propia

En el modelo matemático de una imagen, cada pixel es identificado por el centro de la celdilla formada. La forma de representar un pixel es a través de sus coordenadas (x,y) como un punto en el plano cartesiano. Dependiendo de los distintos tipos de mallado, la distribución de los pixeles cambia.

Figura 6: Pixeles dentro el mallado, de izquierda a derecha. Mallado triangular, Mallado cuadrangular y Mallado hexagonal

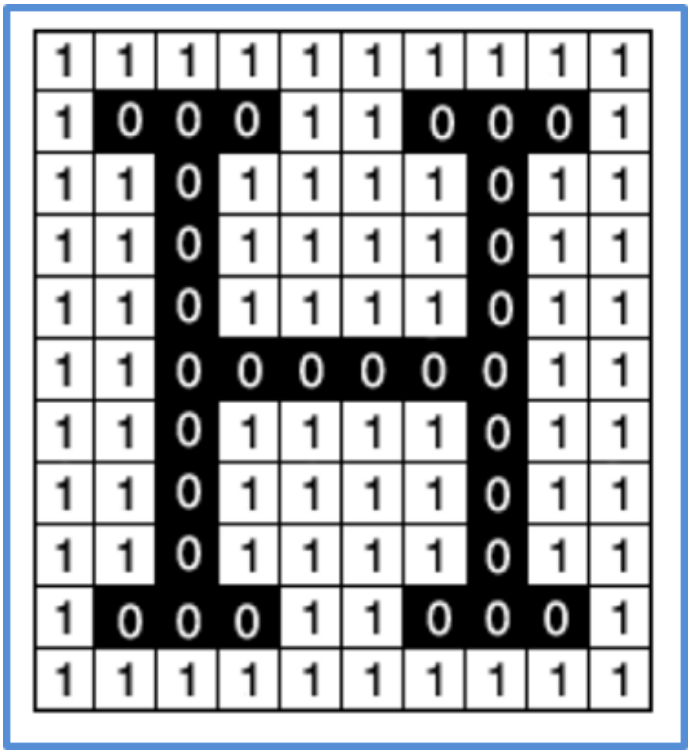


Fuente: Elaboración propia

Para finalizar el proceso de digitalización, se procede con la cuantificación que tiene por objetivo otorgar un valor a cada pixel. El valor puede ser bien un valor

único (escala de grises) o un vector con tres valores por celda (RGB) que define la intensidad de los colores rojo, verde y azul. El rango de valores permite hasta 256 valores que equivale a 8 bits. Las imágenes en escala de grises con solo dos valores en el color: 0 y 1, se denominan imágenes binarias. A continuación un ejemplo de imagen binaria:

Figura 7: Imagen bi-tonal o binaria



Fuente: (Biblioteca de la Universidad de Cornell / Departamento de Investigación, 2003)

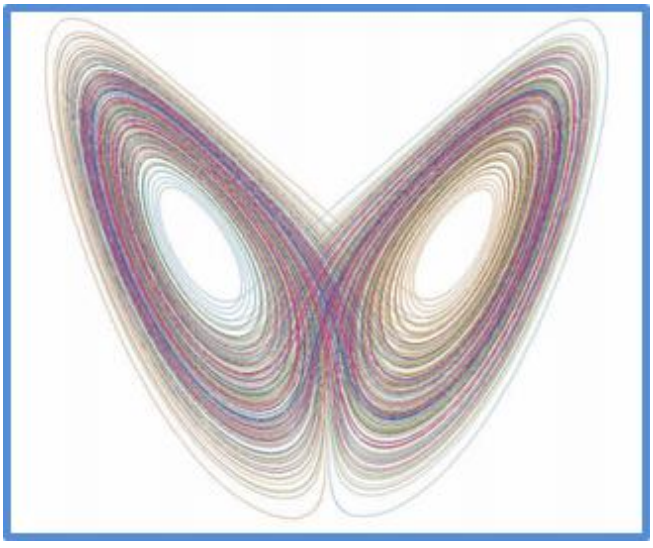
3.2. TEORÍA DEL CAOS

La teoría del caos fue aplicada años después de su descubrimiento por varias áreas de investigación como por ejemplo matemáticas, física, ingeniería, biología, economía y filosofía. Sin importar el área de aplicación, caos significa un estado de desorden. (Ephin, Joy, & Vasanthi, 2013). A continuación se describe las propiedades que acompañan a este tipo de estado que son de gran aporte a los objetivos que busca el presente trabajo.

3.2.1. Historia

La teoría del caos fue descubierta por Edward Norton Lorenz en 1961, más precisamente en instalaciones del MIT[17]. Durante esos días de invierno, Lorenz se encontraba realizando pruebas sobre un modelo climatológico que permitiera predecir el tiempo en la atmósfera basado en doce ecuaciones diferenciales que representaban los parámetros climatológicos. Durante la ejecución del modelo, el resultado fue una gráfica que ahora se conoce como “Atractor de Lorenz”. En base a ese resultado, extraño para él en un principio, decidió reiniciar la ejecución del modelo con un pequeño cambio: las condiciones iniciales tenían seis dígitos decimales y Lorenz los redondeó a tres dígitos para ahorrar tiempo y espacio de procesamiento. Él esperaba razonablemente que la segunda ejecución del modelo sea precisamente igual a la primera, pero no fue así. De hecho, fue casi preciso en un inicio pero después existieron muchas divergencias radicales. En un principio, Lorenz pensó que era debido a un problema de hardware, pero el ordenador Royal McBee utilizado en este modelo, no presentaba problema alguno. Finalmente Lorenz descubrió que el redondeo en las condiciones iniciales, es decir un pequeño cambio en la entrada, produjo resultados totalmente distintos. (Sprott, 2015)

Figura 8: Atractor de Lorenz.



Fuente: (Sprott, 2015)

[17] Massachusetts Institute of Technology

Básicamente, el éxito de Lorenz fue el descubrimiento inesperado de una dependencia sensible sobre las condiciones iniciales, al cual él lo denominó "efecto mariposa". En los sistemas caóticos, la trayectoria se mueve alrededor del atractor conforme pasa el tiempo, sin embargo, dos puntos cercanos en un inicio pueden quedar al finalizar completamente separados por una función exponencial. A pesar que el futuro es determinado únicamente por las ecuaciones que lo gobiernan, pequeñas diferencias en puntos de partida significan grandes diferencias en las condiciones futuras. (Lorenz, 2005).

3.2.2. El caos

Caos constituye una palabra que tiene significados imprecisos en el lenguaje corriente, pero en la ciencia está definida. La experimentación de Lorenz demostró que simplemente tres variables iniciales podrían generar que las predicciones del clima se torne caótico, es decir complicada e impredecible[18] . Sin embargo, este tipo de caos es distinto al comportamiento al azar que comúnmente se define ya que existe cierto orden demostrable geométricamente. Por lo tanto, se puede definir al caos como un efecto impredecible, pero determinable; es decir, el caos no es puramente aleatorio, sino que tiene un orden subyacente. Incluso en cierto sentido, el caos puede llegar a ser determinista.

La teoría postula al mundo como un sistema no lineal[19], más por el contrario, se comporta de manera caótica debido a que una variación en el sistema o en un punto del mismo provoca que, en un lapso de tiempo a futuro, éste presente un comportamiento totalmente diferente e impredecible. La marcada dependencia en las condiciones iniciales genera un comportamiento aparentemente aleatorio propio de los sistemas dinámicos. Por lo tanto, surge el concepto de teoría del caos como una rama de las ciencias exactas que se enfoca en los comportamientos aparentemente aleatorios en los sistemas dinámicos[20].

[18] Cambios no periódicos y crecimiento del efecto de las pequeñas diferencias en el inicio.
[19] Sistema que no sigue un patrón fijo y no es previsible.
[20] Sistemas complejos que cambian o evolucionan con el estado del tiempo.

3.2.3. Sistemas dinámicos

La teoría del caos es considerada como un derivado del campo de la dinámica no lineal la cual ha sido extensamente estudiada. El principio básico del cifrado de imágenes a través del caos está basado en la habilidad de algunos sistemas dinámicos para generar secuencias de números de forma aleatoria. (Philip & Das, 2011).

3.2.3.1. Definición

Según (Alligood, Sauer, & Yorke, 2000), un sistema dinámico consiste en un conjunto de posibles estados y junto a una regla determinan el estado actual en base al estado pasado. Dicha regla debe ser determinística, lo que significa que se puede determinar el estado actual, únicamente en base a los estados anteriores. Existen dos tipos de sistemas dinámicos, de tiempo discreto y tiempo continuo. El trabajo está enfocado a los sistemas dinámicos de tiempo discreto los cuales se caracterizan por considerar el estado actual como entrada al sistema, posteriormente realiza la actualización de la situación y finalmente produce un nuevo estado como salida. Cuando se refiere a "estado" del sistema, significa cualquier información necesaria y de esa manera la regla pueda ser aplicada. Estos sistemas dinámicos discretos son también denominados mapas.

3.2.3.2. Espacio de fases y órbitas

Siendo formales, la definición de sistema dinámico discreto es la siguiente: (Ott, 2002). Sea $X \subset \mathbb{R}^n$ un conjunto no vacío. Sea $f : X \to X$ una función que relaciona estados mediante la expresión:

$$x_{k+1} = f(x_k)$$

La pareja (f, X) es un sistema dinámico. Al conjunto X se le denomina espacio de fases del sistema. En otras palabras, la regla que da la evolución, relaciona los puntos del espacio de fases de manera discreta. Si el sistema inicia en una

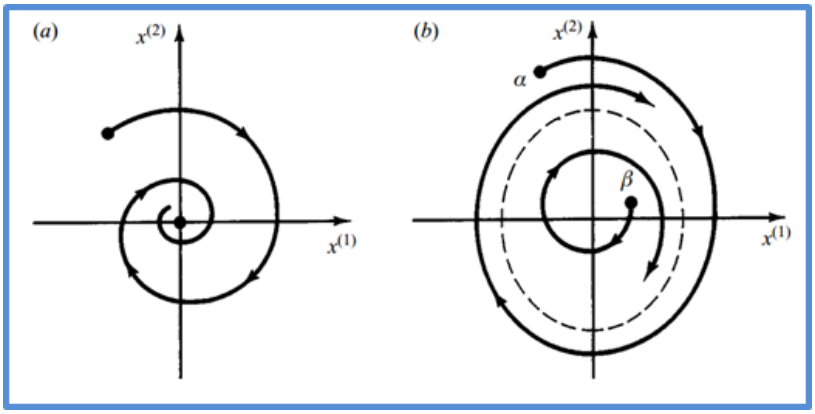
configuración x_0 se le conoce como condición inicial y al conjunto $\{x_0, f(x_0), f^2(x_0), \ldots\}$ órbita o trayectoria de x_0.

3.2.3.3. Atractores

El concepto de atractores es muy importante dentro del estudio de los sistemas dinámicos, específicamente en el espacio de fases. El volumen de espacio de fases de las regiones generadas a partir de las condiciones iniciales describe una asíntota conforme incrementa el tiempo sobre un subconjunto delimitado llamado atractor. En otras palabras, la trayectoria del espacio de fases no es definida, sino que esta errabunda alrededor de un movimiento bien definido; esto significa que el sistema es atraído por un tipo de movimiento generado por un atractor. (Ott, 2002)

Geométricamente, un atractor puede ser un punto, una curva o una estructura fractal. Un ejemplo de atractor puede ser el que describe un oscilador armónico de amortiguación, en la que se puede observar que conforme pase el tiempo, la órbita gira sobre el origen y esta regla se cumple para cualquier condición inicial. Por lo tanto en este caso, el punto origen 0 es llamado atractor del sistema dinámico. Un segundo ejemplo muestra el caso de un ciclo limitado en la que las condiciones iniciales (α) fuera del ciclo limitado describe una órbita una trayectoria alrededor del atractor. El mismo efecto se produce para una condición inicial dentro del ciclo limitado (β) (Ott, 2002)

Figura 9: Ejemplos de atractores



Fuente: (Ott, 2002)

3.2.3.4. Clasificación

Utilizando los conceptos aprendidos de espacio de fases, órbita, y atractores; es posible entender la manera en que los sistemas dinámicos se clasifican:

- Estables: Son aquellos sistemas que tienden a lo largo del tiempo a un punto u órbita. Sus ecuaciones características, condiciones iniciales, sus límites, elementos y relaciones nos permiten conocer su evolución a través del tiempo, es decir, es posible determinar hacia donde lo dirige su atractor.
- Inestables: Los sistemas inestables, en cambio, no se guían por atractores, se escapan de éstos y no tienden hacia un punto. Una de sus características es la alta dependencia a sus condiciones iniciales.
- Caóticos: Los sistemas caóticos, por su parte, manifiestan ambos comportamientos. Es decir, existe un atractor por el que el sistema se ve atraído, pero a la vez, hay "fuerzas" que lo alejan de éste. De esa manera, el sistema permanece confinado en una zona de su espacio de estados, pero sin tender a un atractor fijo. Las trayectorias correspondientes a condiciones iniciales tan vecinas como se quiera, divergen de manera exponencial con el tiempo. En los sistemas caóticos de pueden conocer sus ecuaciones y sus condiciones iniciales fijas, sin embargo la más mínima variación provoca una evolución radical en su comportamiento. (Alligood, Sauer, & Yorke, 2000)

Dado el enfoque y el alcance del presente trabajo, los sistemas dinámicos caóticos serán el punto de atención a lo largo del documento.

3.2.4. Mapas caóticos

Los sistemas dinámicos caóticos son descritos por mapas que surgen como modelos de fenómenos naturales casi realistas. En otras palabras, la ciencia busca predecir la manera en la que un sistema evolucionará conforme pase el tiempo. Un claro ejemplo de esta inquietud se enfoca en la evolución de la población. Una función tan simple como $f(x) = 2x$ es una regla que asigna a

cada valor de x, un valor doblemente mayor. La evolución de este proceso dinámico es descrita por la composición de la función f. Se define $f^2(x) = f\big(f(x)\big)$ y de manera general $f^k(x)$ como el resultado de aplicar la función f al estado inicial k veces. Es decir, dado un valor inicial de x, se requiere conocer el resultado de $f^k(x)$ en el tiempo k. En este ejemplo está claro que si el valor inicial de x es mayor a cero, la población se incrementará sin ningún límite. Este tipo de expansión, en el cual la población es multiplicada una constante por cada unidad en el tiempo, se denomina crecimiento exponencial. (Alligood, Sauer, & Yorke, 2000). En base a este ejemplo se describe el concepto de mapa y como llega a adquirir un comportamiento caótico.

3.2.4.1. Mapas

El hecho que los habitantes tengan recursos limitados no concuerda con el concepto del crecimiento exponencial de la población. Es decir, el crecimiento de la población como resultado de la multiplicación por una constante no puede continuar por siempre. En algún momento los recursos del entorno estarán comprometidos con el incremento de habitantes lo que causará que el crecimiento reduzca de una forma menor que la exponencial.

En otras palabras, aunque la regla $f(x) = 2x$ puede ser correcta para cierto rango de la población, puede perder aplicabilidad en otros rangos. Un modelo improvisado puede ser utilizado para la población con recursos limitados, la misma está dada por $g(x) = 2x(1 - x)$, donde x es medido en millones. Cuando la población es pequeña, el factor $(1 - x)$ se acerca a uno y $g(x)$ se asemeja a la función $f(x)$. Este es el efecto no lineal y el modelo dado por $g(x)$ es un ejemplo de modelo logístico.

Para verificar los resultados obtenidos y establecer las diferencias, se realiza el cálculo respectivo sobre las funciones $f(x)$ y $g(x)$. Se inicia con un valor pequeño como $x = 0.01$ y se calcula $f^k(x)$ y $g^k(x)$ para los siguientes valores de k. Los

resultados del modelo se muestran en la siguiente tabla, en la que para $g(x)$ existe una evidencia de que la población alcanza eventualmente a un tamaño límite que se denomina estado estable de la población para el modelo $g(x)$.

Tabla 2: Comparación del modelo exponencial de crecimiento $\boldsymbol{f(x) = 2x}$ al modelo logístico $\boldsymbol{g(x) = 2x(1 - x)}$

k	$f^k(x)$	$g^k(x)$
0	0.0100000000	0.0100000000
1	0.0200000000	0.0198000000
2	0.0400000000	0.0388159200
3	0.0800000000	0.0746184887
4	0.1600000000	0.1381011397
5	0.3200000000	0.2380584298
6	0.6400000000	0.3627732276
7	1.2800000000	0.4623376259
8	2.5600000000	0.4971630912
9	5.1200000000	0.4999839039
10	10.2400000000	0.4999999995
11	20.4800000000	0.5000000000
12	40.9600000000	0.5000000000

Fuente: (Alligood, Sauer, & Yorke, 2000)

Existen marcadas diferencias entre el comportamiento de la cantidad de población descritas por los modelos $f(x)$ y $g(x)$. Bajo el sistema dinámico $f(x)$, la cantidad inicial de población $x = 0.01$ genera un resultado arbitrariamente cuantioso conforme pasa el tiempo. Mientras que bajo el sistema $g(x)$, la misma cantidad inicial de población $x = 0.01$ genera en un principio un comportamiento similar. Sin embargo, la cantidad límite es alcanzada y nunca más vuelve a cambiar.

En conclusión, una función cuyo espacio de dominio (input) y espacio de rango (output) son iguales, se considera un mapa. Sea x un punto y f un mapa. La órbita de x dado f es el conjunto de puntos $\{x, f(x), f^2(x), \ldots\}$. El punto inicial x para la órbita es llamado valor inicial dela órbita. Un punto p es un punto ajustado del mapa f si $f(p) = p$. Para el ejemplo de la función $g(x) = 2x(1-x)$ que es un mapa, la órbita de $x = 0.01$ bajo la función g es $\{0.01, 0.0198, 0.0388, \ldots\}$ y los puntos ajustados de g son $x = 0$ y $x = {}^{1}/_{2}$. (Alligood, Sauer, & Yorke, 2000)

3.2.4.2. Clasificación

De acuerdo a una recopilación de algunos ejemplos de mapas caóticos, a continuación se detallan algunos de ellos con sus respectivos atributos tomando en cuenta que actúan bajo el dominio del tiempo discreto:

Tabla 3: Clasificación de mapas caóticos en tiempo discreto

Mapas Caóticos	Dominio de Espacio		Número de Dimensiones	
	Real	Complejo	Una	Dos
Arnold's cat map	X			X
Baker's map	X			X
Circle map	X		X	
Complex quadratic map		X	X	
Complex squaring map		X	X	
Duffing map	X			X
Dyadic transformation	X		X	
Exponential map		X		X
Gauss map	X		X	
Gingerbreadman map	X			X

Hénon map	X			X
Horseshoe map	X			X
Ikeda map	X			X
Interval exchange map	X		X	
Kaplan-Yorke map	X			X
Logistic map	X		X	
Lozi map	X			X
Pomeau-Manneville maps for intermittent chaos	X		X	X
Shobu-Ose-Mori piecewise-linear map	X		X	
Standard map, Kicked rotor	X			X
Tent map	X		X	
Tinkerbell map	X			X
Zaslavskii map	X			X

Fuente: Elaboración propia

Debido a la diversidad de mapas caóticos en tiempo discreto que fueron desarrollados por los investigadores, el presente trabajo se enfoca en aquellos mapas unidimensionales debido a su simplicidad y alto nivel de eficiencia como ventajas subyacentes para los propósitos deseados. (Gao, Zhang, Liang, & Li, 2006) (Delfs & Knebl, 2006)

3.2.4.3. Mapas Unidimensionales

El presente trabajo se enfoca en el comportamiento unidimensional de los sistemas dinámicos en los que la variable del tiempo es discreta, es decir,

ecuaciones en diferencias finitas, relaciones de recursión, mapas iterados, mapas dinámicos o simplemente mapas. Las razones por las cuales se ha elegido este tipo de mapas son las siguientes (Thunberg, 2001):

- Sistemas de mayores dimensiones pueden ser reducidos, algunas veces, en sistemas de una sola dimensión. Esta práctica es posible considerando mapas resultantes de sistemas de tiempo continuo en subconjuntos de sistemas de menor dimensión.
- Para mucho de los resultados expuestos en el artículo (Thunberg, 2001), existen relaciones análogas para sistemas mucho más complejos, algunos de ellos comprobados y otros con la expectativa de ser posible representarlos de manera genérica. Por tanto, entender sistemas de una sola dimensión da lugar a aplicar un modelo de modelos para alcanzar sistemas más complejos.

Por ejemplo, $x_{n+1} = \cos x_n$ es un mapa unidimensional puesto que los puntos x_n pertenecen al espacio unidimensional de los números reales. La forma general de mapa unidimensional se describe de la siguiente forma: (Belkhouche & Qidwai , 2003)

$$x_{n+1} = f(x_n, r), \text{ dado un valor inicial } x_0 \text{ y una constante } r$$

Para generar las iteraciones de un mapa unidimensional, se puede hacer uso de la representación gráfica. Por ejemplo, se utiliza el siguiente mapa unidimensional:

$$x_{n+1} = 2.8x_n(1 - x_n), \text{ dado un valor inicial } x_0 = 0.1$$

Las iteraciones son las siguientes:

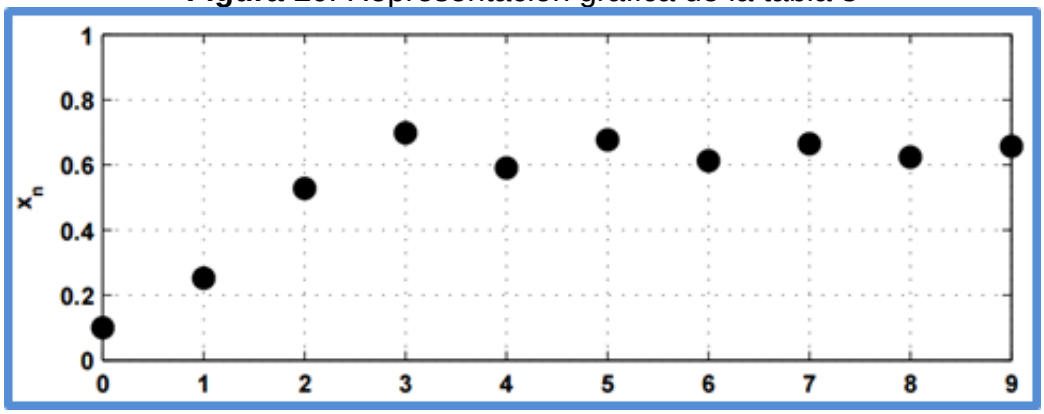
Tabla 4: Iteraciones de un mapa unidimensional con $\boldsymbol{r = 2.8}$ y $\boldsymbol{x_0 = 0.1}$

n	x_n
0	0.1
1	0.2520
2	0.5278
3	0.6978
4	0.5904
5	0.6771
6	0.6122
7	0.6648
8	0.6240
9	0.6570

Fuente: Elaboración propia

La representación gráfica es la siguiente:

Figura 10: Representación gráfica de la tabla 3



Fuente: (Weckesser, 2005)

Los mapas unidimensionales surgen como representaciones discretas de ecuaciones diferenciales para facilitar el estudio numérico a través de ordenadores digitales que son más adecuados para tratar variables discretas que continuas. En

el presente trabajo se hace uso de este concepto para estudiar el caos y complejidad dado que el estado de un sistema va saltando a lo largo del su trayectoria en lugar de fluir continuamente, son capaces de lograr un comportamiento caótico.

3.2.5. Mapa logístico

Como se ha mencionado anteriormente, el presente trabajo se enfoca en el mapa logístico que viene dado por un sistema dinámico muy simple en apariencia y está representado por una ecuación logística. Creado por el biólogo Rober May que utilizó para el estudio de la evolución de la población de insectos en un sistema cerrado. (Strien & Melo, 2012).

3.2.5.1. Definición

Básicamente, este sistema es discreto y además unidimensional y viene dado por la siguiente fórmula:

$$x_{k+1} = \mu x_k(1 - x_k)$$

Donde la constante μ oscila entre $0 < \mu < 4$. El espacio de fases del sistema es en el intervalo $[0,1]$

3.2.5.2. Comportamiento caótico

Incluso en sistemas de una sola dimensión, es posible generar comportamientos caóticos y para demostrar este efecto es necesario conocer los puntos fijos y periódicos. En este caso, se describe los mismos para la ecuación logística descrita en el punto anterior.

$$x_{k+1} = \mu x_k(1 - x_k)$$

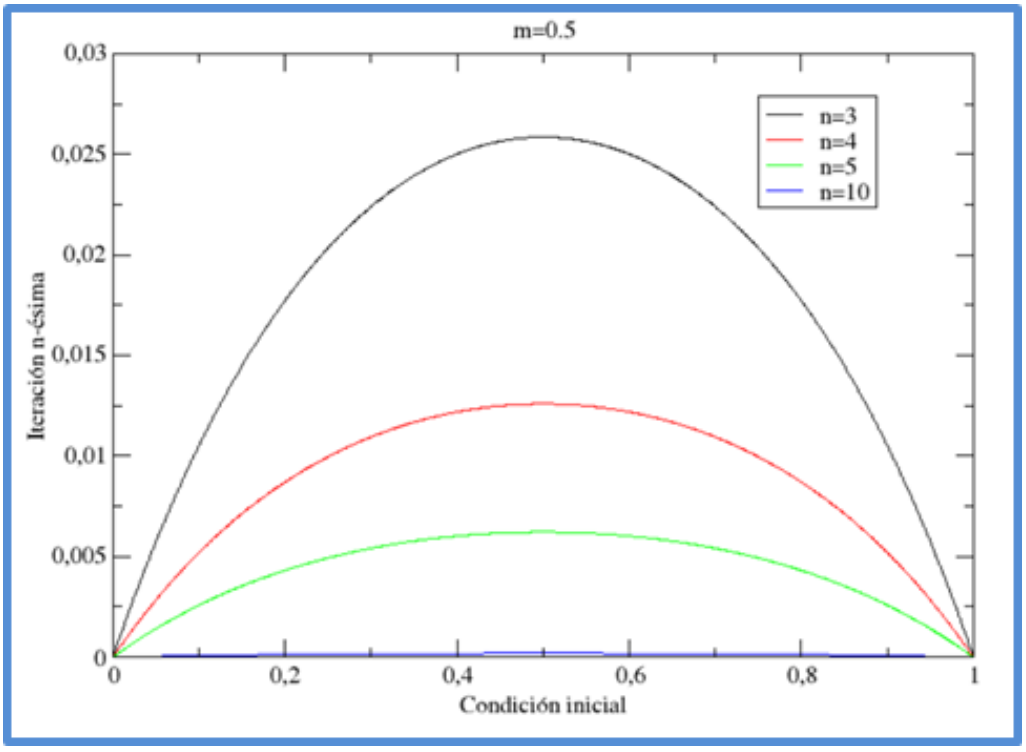
Los puntos fijos deben cumplir la regla $x_{k+1} = x_k$. En este sentido y tras operaciones matemáticas se comprueba que las soluciones son:

$$p_0 = 0 \qquad \text{y} \qquad p_\mu = \frac{\mu - 1}{\mu}$$

3.2.5.2.1. Caso $0 \leq \mu \leq 1$

Si $\mu = 0$, entonces el sistema alberga una tendencia hacia 0 que es el único punto fijo. Por otro lado, si $0 < \mu < 1$, entonces la solución no será válida debido a que predice puntos fijos que no pertenecen al espacio de fases; por lo cual la solución vuelve a ser $x = 0$ como el único punto fijo. Finalmente, si $\mu = 1$, la segunda solución predice como resultado $x = 0$.

Figura 11: Evolución temporal de la ecuación logística en intervalo $\mathbf{0 \leq \mu \leq 1}$



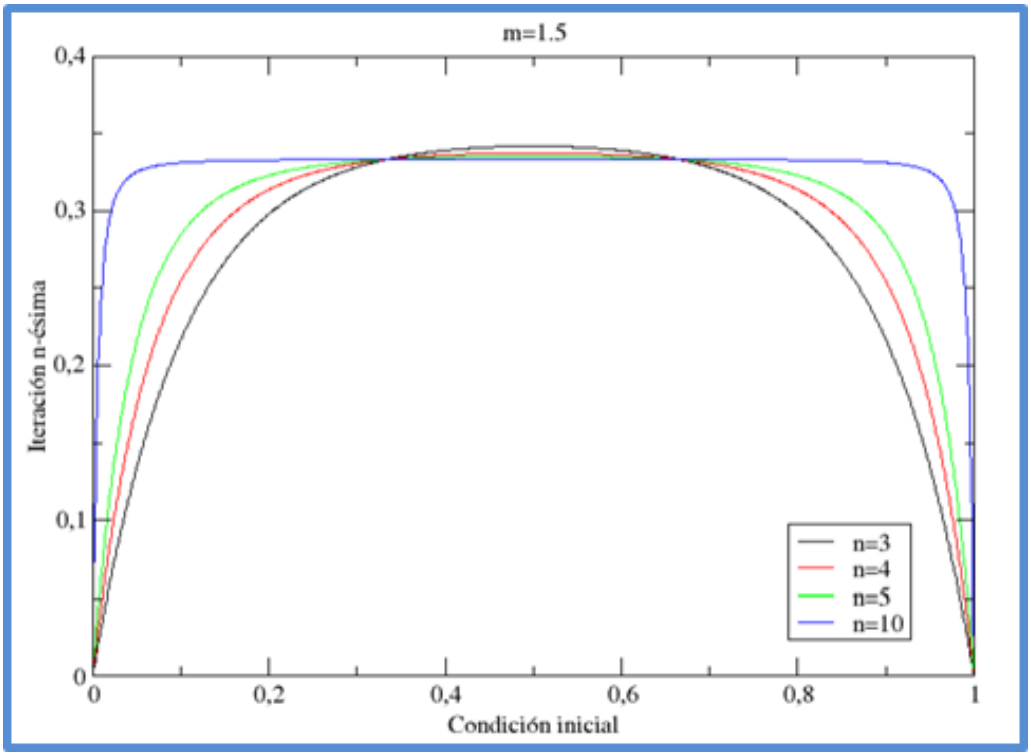
Fuente: (Alligood, Sauer, & Yorke, 2000)

En la anterior figura se observa la evolución temporal para cada una de las condiciones iniciales dentro el intervalo $[0,1]$ con el parámetro $\mu = 0.5$. Se observa la manera en que las órbitas tienden rápidamente al punto fijo $x = 0$.

3.2.5.2.2. Caso $1 < \mu \leq 3$

En este caso, $p_\mu \in [0,1]$ por tanto se tiene dos puntos fijos. Al igual que el anterior caso, se calcula la evolución n-ésima para cada punto del intervalo $[0,1]$, obteniendo la siguiente figura:

Figura 12: Evolución temporal de la ecuación logística en intervalo $1 < \mu \leq 3$



Fuente: (Alligood, Sauer, & Yorke, 2000)

En la imagen anterior, se traza la evolución temporal para cada una de las condiciones iniciales dentro el intervalo $[0,1]$ con el parámetro $\mu = 1.5$. Se observa la manera en que las órbitas tienden rápidamente al punto $x = 0.334$.

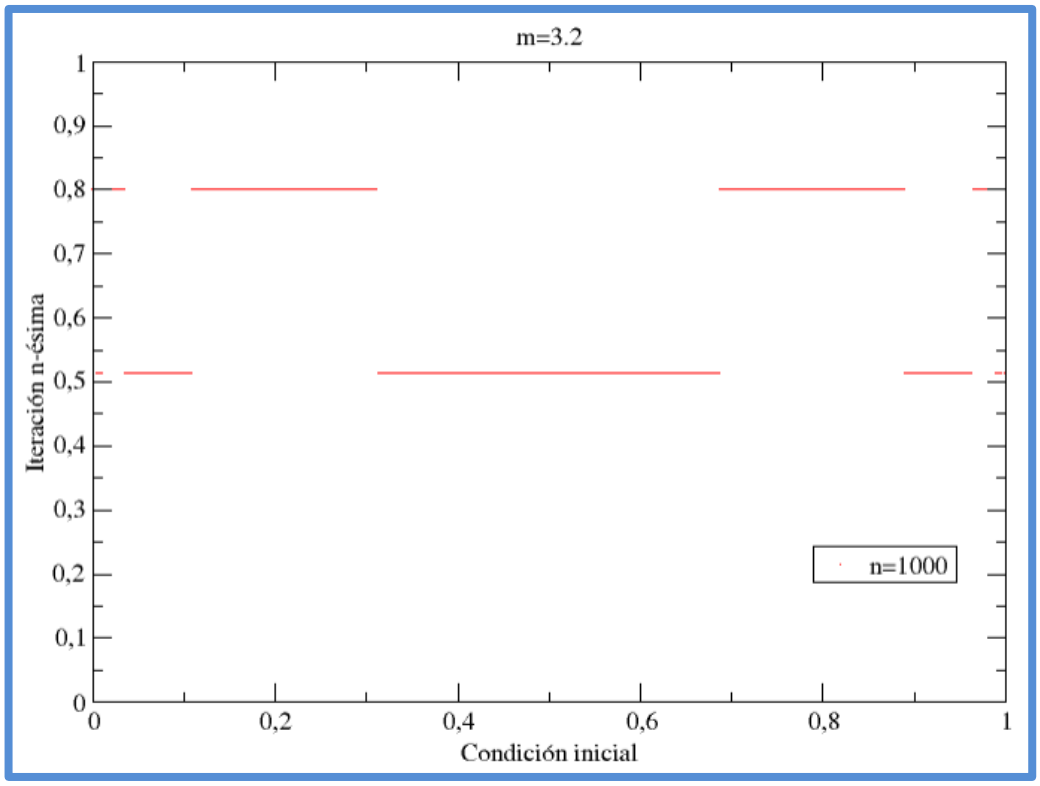
3.2.5.2.3. Caso $3 < \mu \leq 4$

Al momento no ha se ha generado en ningún momento el Caos, mas ahora en este intervalo es dónde cobra sentido el uso de este concepto en la criptografía. En este intervalo, los puntos $x = 0$ y $x = p_u$ continúan siendo fijos, eso no cambia

en este intervalo. Sin embargo, de la misma forma que el intervalo anterior, ambos vuelven a ser puntos fijos repulsivos[21]. Por lo tanto, los dos únicos puntos fijos del sistema son repulsivos y por ende no tiende a un punto fijo atractivo, pues no lo hay.

Al realizar una simulación con 1000 iteraciones, se demuestra que no alcanza el equilibrio y en la siguiente figura en la que se muestran dos puntos a los que la solución tiende alternativamente con lo que se genera de puntos periódicos de periodo dos.

Figura 13: Evolución temporal de la ecuación logística en intervalo $\mathbf{3 < \mu \leq 4}$



Fuente: (Alligood, Sauer, & Yorke, 2000)

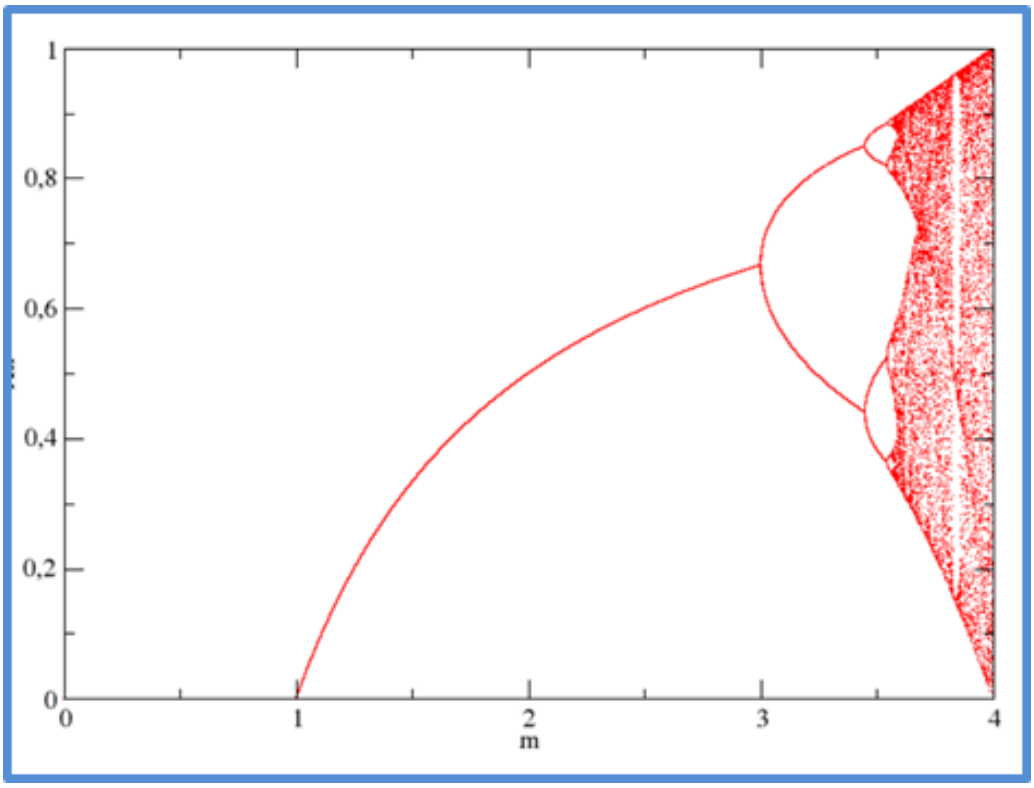
En la imagen anterior, se traza la evolución temporal para cada una de las condiciones iniciales dentro el intervalo $[0,1]$ con el parámetro $\mu = 3.2$. Se

[21] Es un punto que una pequeña perturbación de la posición saca rápidamente al sistema de su configuración.

observa que la evolución del sistema no tiende a un punto fijo como antes, sino que pareciera que tiende a dos puntos diferentes por igual.

Se demuestra que para el punto $\mu = 3$, el punto p_μ pasa de ser fijo atractivo a ser fijo repulsivo dividiéndose en dos puntos atractivos de periodo dos, lo que genera el fenómeno denominado "duplicación de periodo". Este efecto tiende a incrementar cuando aumenta μ, ya que estos dos puntos dan lugar a otros cuatro puntos periódicos atractivos de periodo cuatro. Generalizando, antes de llegar al valor $\mu = 4$, los resultados empiezan a doblarse y doblarse de tal manera que llega a un punto en el que el número de puntos periódicos en el intervalo $[0,1]$ es muy denso llegando a generar un sistema caótico. La representación de este efecto se denomina figura de Feigenbaum y se muestra a continuación:

Figura 14: Gráfico de Feigenbaum



Fuente: (Alligood, Sauer, & Yorke, 2000)

El aspecto fractal de la figura denota uno de los aspectos de la aparición del Caos, la multiplicación de los puntos periódicos del sistema. Se observa como el intervalo $[0,1]$, el sistema tiende a 0; el intervalo $[1,3]$ tiende a un número distinto de 0 y que varía con μ. Al pasar por el punto 3, aparecen dos puntos periódicos, pero pronto se observa que estos a su vez se subdividen en más adelante. Claramente, antes de llegar a 4, el sistema se ha descontrolado y es impredecible por tanto el Caos se ha generado. (Alligood, Sauer, & Yorke, 2000)

Debido a la aplicación de la teoría del caos sobre criptografía, se describen a continuación las propiedades más importantes de los sistemas caóticos para tal efecto:

- Alta sensibilidad a condiciones iniciales
- Ergodicidad
- Espectro de banda ancha
- Pseudo-aleatorio
- Sin periodicidad

3.3. SEGURIDAD Y CRIPTOGRAFÍA

En las últimas décadas, varios investigadores dieron cuenta de una fuerte relación entre caos y criptografía. En realidad, el caos y el ruido en sistemas reales, son dos comportamientos válidos e irregulares y por tanto el uso de ellos en criptografía es de igual manera válido. Una de las ventajas del sistema caótico visto en el anterior punto (apartado 2.2) es la naturaleza determinística la cual facilita al proceso de desencriptación. (Pisarchik & Zanin, 2008). Todos estos antecedentes hicieron que las técnicas criptográficas se dividan en dos grupos en base al enfoque: basadas en caos y no basadas en caos. (Sankpal & Vijaya, 2014). A continuación se describe el cifrado o encriptación caótica en conjunto con conceptos teóricos que relacionan ambas directrices del trabajo (caos y seguridad) y su aplicación en el objeto de estudio (imágenes digitales).

3.3.1. Definición

Criptografía es la ciencia que se encarga de proteger la privacidad de la información durante una comunicación bajo condiciones hostiles.

3.3.2. Cifrado de imágenes

El cifrado puede ser definido como la conversión de un texto plano en un texto cifrado que no puede ser entendida por ninguna persona sin haber desencriptado previamente el texto encriptado. La desencriptación es el proceso inverso al cifrado que consiste en retornar al estado original del texto (texto en plano) de tal manera que pueda ser entendida.

3.3.3. Criptografía caótica

En 1989, (Matthews, 1989) propuso por primera vez la aplicación de conceptos propios de la teoría del caos dentro de la algoritmia dirigida a la criptografía. La sensibilidad en la dependencia sobre las condiciones iniciales es el resultado del comportamiento randómico que ofrecen los sistemas dinámicos caótico por su naturaleza determinística. Debido a que estos sistemas no pueden ser especificados con alta precisión, el comportamiento de los sistemas caóticos se torna tan impredecible que se asemeja al ruido. Esto supone una estrecha relación entre criptografía y caos que convierte a los algoritmos criptográficos basados en caos un candidato ideal para securizar no solo recursos, sino también canales de comunicación.

Los algoritmos criptográficos y los mapas caóticos guardan similitudes y diferencias. Algunas propiedades similares entre ambos son: sensibilidad a cambios en las condiciones iniciales y control de parámetros, comportamiento pseudoaleatorios y orbitas periódicas inestables con periodos largos. El principio básico en el cifrado de imágenes a través del caos se basa en la habilidad de algunos sistemas dinámicos de producir una secuencia de números que son aleatorios por naturaleza. Una diferencia importante entre ambos conceptos es que las transformaciones del cifrado son definidas por conjuntos finitos, mientras

que los mapas caóticos aplican solamente para números reales. A continuación se listan las diferencias y similitudes entre ambos conceptos: (Sankpal & Vijaya, 2014)

Tabla 5: Similitudes y Diferencias entre Caos y Criptografía

Sistemas Caóticos	Algoritmos Criptográficos
Espacio de Fase: Conjunto de números reales	Espacio de Fase: Finito. Conjunto de números enteros
Iteraciones	Rondas
Parámetros	Llaves
Sensible a condiciones iniciales y control de parámetros	Difusión

Fuente: (Sankpal & Vijaya, 2014)

3.3.4. Confusión y Difusión

Muchas características del caos, como la ergodicidad, espectro de banda ancha, la alta sensibilidad a condiciones iniciales, etc., están conectadas directamente con las dos propiedades básicas de los cifradores: confusión y difusión. (Zhang, Liao, & Wang, 2005)

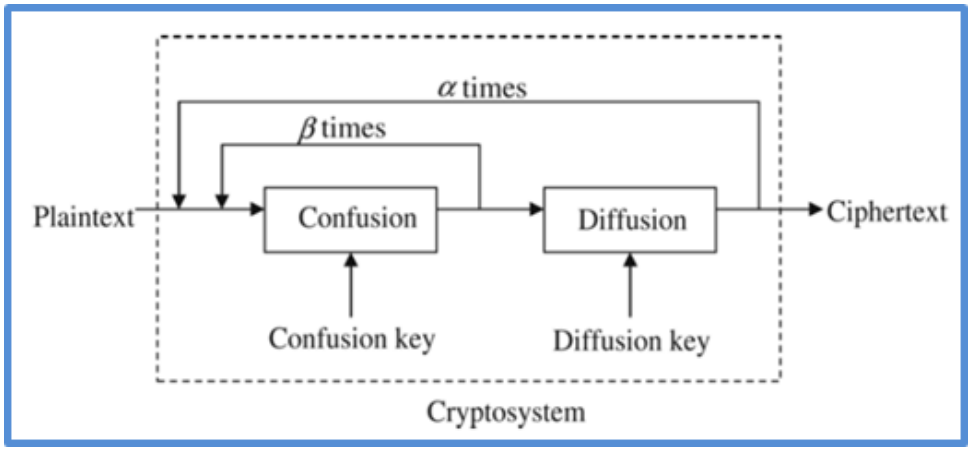
La etapa de confusión consiste básicamente en el intercambio de posiciones de los pixeles de tal manera que queden mezclados en toda la imagen digital sin alterar el valor de los pixeles. Para realizar este proceso se requiere de ciertas condiciones iniciales y parámetros de control que en conjunto conforman la llave secreta. Aunque a través de esta etapa la imagen queda irreconocible, no es muy seguro dejarlo en este estado ya que podría ser descifrado por cualquier ataque. En ese entendido y para mejorar la seguridad, la segunda etapa del proceso de encriptación apunta a cambiar el valor de cada pixel de la imagen entera.

El proceso de difusión es llevado a cabo a través del mapa caótico el cual es principalmente dependiente de las condiciones iniciales y los parámetros de control. En esta etapa, el valor los pixeles es modificado secuencialmente por

medio de una secuencia generada por el sistema caótico. (Sankpal & Vijaya, 2014)

El proceso entero de confusión-difusión se repite por un número determinado de veces para lograr un nivel satisfactorio de seguridad. Además, la propiedad randómica inherente en los mapas caóticos lo convierte en un proceso adecuado para la encriptación de imágenes digitales. Por lo tanto, debido a que la confusión y difusión suponen un desorden determinístico, un sistema criptográfico tradicional puede ser considerado como un sistema caótico o sistema pseudorandómico. (Pisarchik & Zanin, 2008)

Figura 15: Esquema general de un sistema criptográfico



Fuente: (Pisarchik & Zanin, 2008)

Matemáticamente, el esquema del criptosistema puede ser representado de la siguiente manera: (Lian, Sun, & Wang, 2005)

$$\boldsymbol{R} = D^{\alpha}\left(C^{\beta}(\boldsymbol{P}, K_C), K_D\right)$$

Donde ***P*** y ***R*** representan la imagen en plano y la imagen cifrada respectivamente; *C* y *D* representan a las funciones de confusión y difusión; K_C y K_D corresponden a las llaves secretas tanto de la confusión como de la difusión; finalmente, α y β expresan el número de iteraciones para cada etapa. Mientras mayor sea la sensibilidad de las funciones *C* y *D* con respecto a sus llaves K_C y K_D y mayor el

tamaño de la llave, de igual manera aumentará el nivel de seguridad. El tamaño de la llave secreta se define de la siguiente manera: (Pisarchik & Zanin, 2008)

$$S = \left({S_C}^{\beta} S_D\right)^{\alpha}$$

Donde S_C y S_D son el tamaño de las llaves de confusión y difusión respectivamente y están determinados por el tamaño de llaves de las condiciones iniciales y los parámetros de las funciones de confusión y difusión. Mientras mayor sean las potencias α y β, mayor será el tamaño de la llave del sistema y por tanto mayor la seguridad del mismo. Sin embargo, existe un parámetro a tomar en cuenta llamado EDT[22] el cual también incrementa en la misma magnitud que α y β. Por lo tanto, para diseñar el sistema criptográfico se debe realizar un balance entre la seguridad y la velocidad de procesamiento.

[22] Encryption/Decryption Time

CAPÍTULO IV

TECNOLOGÍAS UTILIZADAS

4.1. OPENCV

Dentro de las tecnologías aplicadas en el presente trabajo, se encuentra OpenCV como herramienta para la manipulación de imágenes digitales basadas en lenguajes de programación de bajo nivel. A continuación se profundiza más en su funcionamiento y conceptos importantes para el trabajo realizado.

4.1.1. Definición

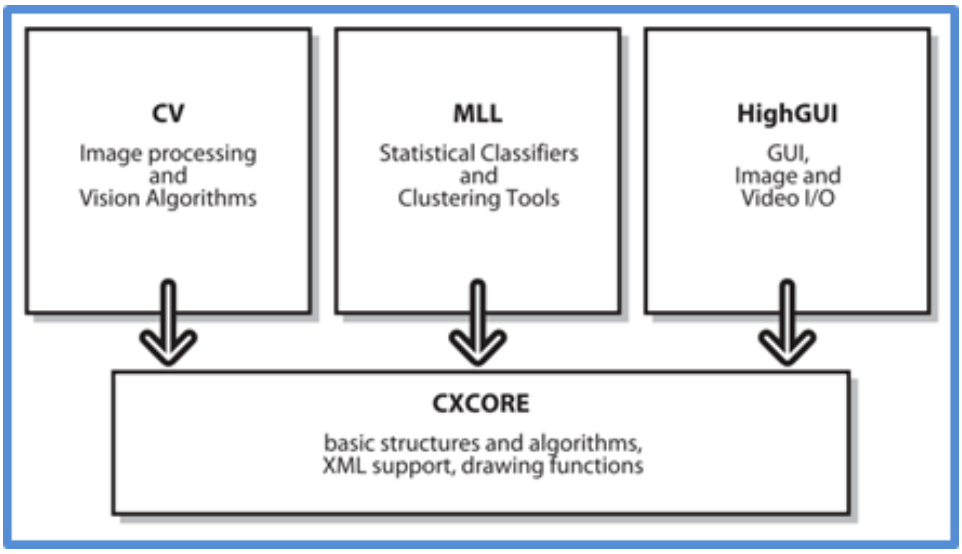
OpenCV es una librería de código abierto para la manipulación de imágenes digitales. La librería está escrita en C y C++ y puede ser ejecutada bajo entornos Linux, Windows y Mac OS X. Actualmente, existe un desarrollo activo para aplicar sobre interfaces como Python, Ruby, Matlab y otros lenguajes. (Bradski & Kaehler, 2008).

OpenCV fue diseñado en pro de la eficiencia computacional y con fuerte enfoque a las aplicaciones de tiempo real. OpenCV está codificado en lenguaje C optimizado aprovechando los procesadores multi-núcleos. Otro de los objetivos de OpenCV es proveer una infraestructura de computación de uso simple que permita construir aplicaciones visuales sofisticadas de manera más rápida. Contiene 500 funciones aplicables a varias áreas de visualización, incluyendo inspección de productos fabricados, imágenes médicas, seguridad, interfaces de usuario, calibración de cámaras y robótica.

4.1.2. Estructura y contenido

OpenCV está compuesto básicamente en cinco principales componentes, cuatro de ellos se muestran en la siguiente imagen. Lo componentes CV contienen el procesamiento de imagen básico y algoritmos de computación grafica de alto nivel; MLL es la librería de máquina de aprendizaje, la cual incluye clasificadores estadísticos y herramientas de clustering. HighGUI contiene rutinas de entrada y salida (I/O) y funciones relacionadas al almacenamiento y carga de videos e imágenes. Finalmente el CXCore contiene la estructura de datos básica y su contenido.

Figura 16: La estructura básica de OpenCV



Fuente: (Bradski & Kaehler, 2008).

4.2. CUDA – GPU

Dentro de las tecnologías aplicadas en el presente trabajo, se encuentra un tipo de procesamiento cuya arquitectura permite acelerar la ejecución de ciertas tareas basadas en imágenes digitales. A continuación se detalla a profundidad conceptos y funcionamiento de esta tecnología.

4.2.1. Definición

CUDA es una arquitectura de cálculo paralelo de NVIDIA que aprovecha la gran potencia de la GPU (unidad de procesamiento gráfico) para proporcionar un incremento extraordinario del rendimiento del sistema. (NVIDIA Corporation, 2015)

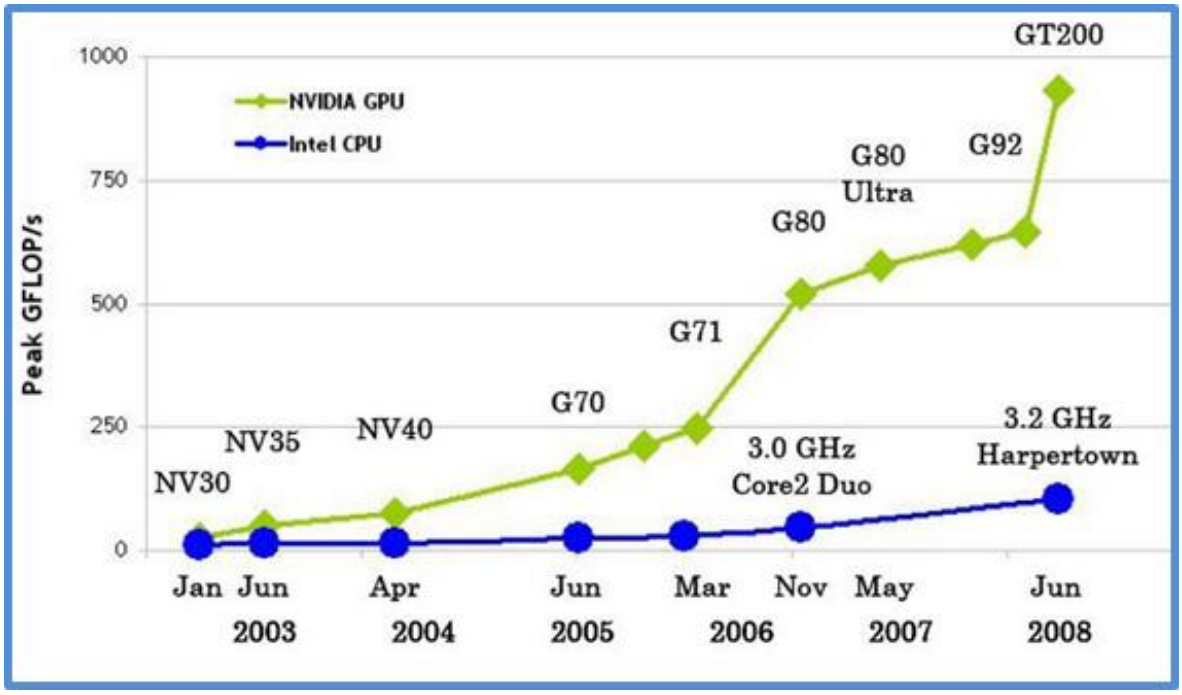
Innumerables aplicaciones prácticas ya han aplicado esta esta tecnología en campos como el procesamiento de vídeo e imágenes, la biología y la química computacional, la simulación de la dinámica de fluidos, la reconstrucción de imágenes de TC, el análisis sísmico o el trazado de rayos, entre otras. (NVIDIA Corporation, 2015). A continuación se explica a profundidad la importancia del uso de estas unidades de procesamiento gráfico.

4.2.2. GPU y procesamiento en paralelo

Los sistemas informáticos están pasando de realizar el "procesamiento central" en la CPU a realizar "co-procesamiento" repartido entre la CPU y la GPU. Para posibilitar este nuevo paradigma computacional, NVIDIA ha inventado la arquitectura de cálculo paralelo CUDA, que ahora se incluye en las GPUs GeForce, ION Quadro y Tesla GPUs, lo cual representa una base instalada considerable para los desarrolladores de aplicaciones. En el presente trabajo se hace uso de la tarjeta de video GPUs GeForce. (NVIDIA Corporation, 2015)

Desde hace un tiempo las GPUs programables han evolucionado como una unidad de gran carga de trabajo y en comparación con el procesamiento en CPU, la diferencia en Gigaflops[23] entre CPUs y GPUs se muestra a continuación:

Figura 17: Diferencias de GigaFlops entre CPUs y GPUs



Fuente: (NVIDIA Corporation, 2015)

[23] Las operaciones de coma flotante por segundo son una medida del rendimiento de una computadora, especialmente en cálculos científicos que requieren un gran uso de operaciones de coma flotante

Para comprender el funcionamiento de una GPU, es necesario conocer conceptos relacionados al vertex shaders y los píxel shaders que se detallan líneas abajo.

4.2.2.1. Vertex Shaders y Píxel Shaders

La palabra “shaders” hace referencia a subprogramas encargados del procesamiento de vértices (vertex shaders) y de pixeles (píxel shaders). Debido a que son pequeños programas, los mismos pueden ser programados por el desarrollador lo que constituye la principal ventaja ya que otorga flexibilidad que antes de su creación no era posible.

Tal como lo indica en la documentación de NVIDIA (NVIDIA Corporation, 2015), un vertex shader es una función que recibe como parámetro un vértice. Una de las características es que sólo trabaja con un vértice a la vez, y no puede eliminarlo, sólo transformarlo. Con esta finalidad, este shader modifica propiedades del mismo con la intención que repercutan en la geometría del objeto al que pertenece. Por otra parte, un píxel shader básicamente especifica el color de un píxel. Este tratamiento individual de los píxeles permite que se realicen cálculos principalmente relacionados con la iluminación del elemento del cual forman parte en la escena, y en tiempo real. (Sánchez, 2008)

La incorporación de los píxel shaders y vertex shaders permite a los programadores una mayor libertad a la hora de manipular los gráficos ya que puede tratarse a cada píxel y cada vértice por separado. De esta manera, la gestión y tratamiento de las imágenes digitales son realizadas mucho más detalladamente, sucediendo lo mismo con la geometría de los objetos.

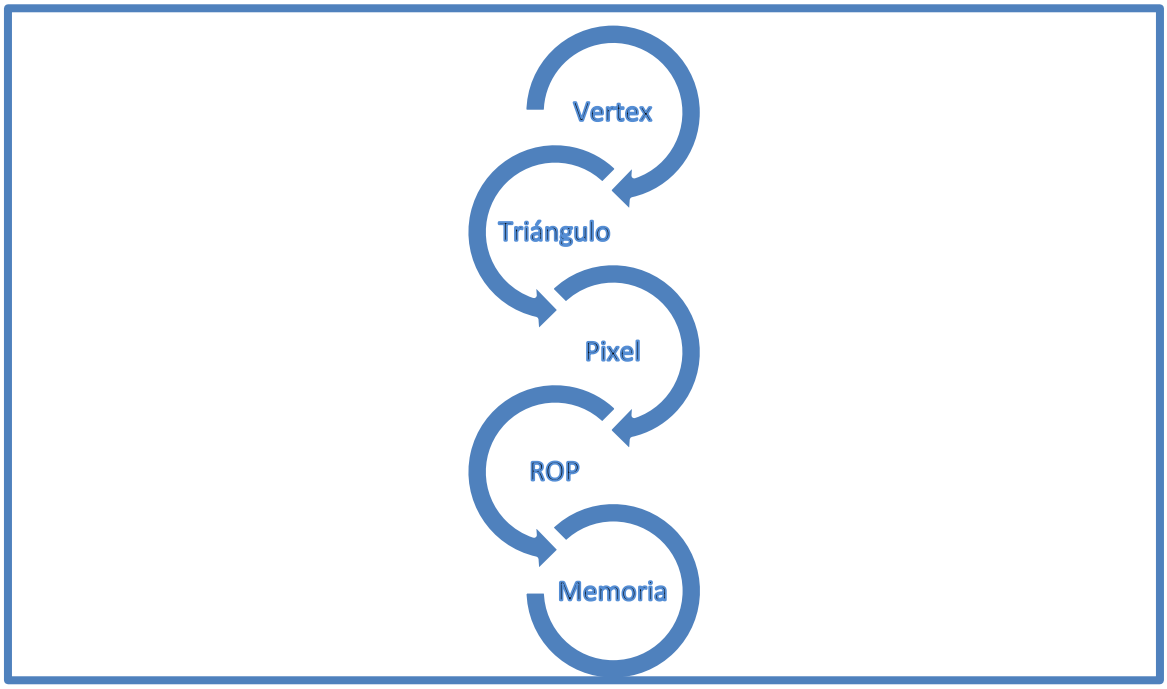
4.2.2.2. Pipeline[24] del procesamiento en GPU

Cuando se revisa las arquitecturas hardware, el flujo de datos, y las operaciones pipeline, a menudo es bueno empezar por el nivel más alto, donde los datos llegan desde la CPU a la GPU, y el proceso se desarrolla hacia abajo a través de

[24] La segmentación es un método por el cual se consigue aumentar el rendimiento de algunos sistemas electrónicos digitales. Es aplicado, sobre todo, en microprocesadores.

múltiples fases de procesamiento hasta que un píxel es manipulado. (Sánchez, 2008). Para situarnos, las GPUs han utilizado diseños pipeline tradicionales, como los que aparecen ilustrados en siguiente figura:

Figura 18: Pipeline clásico de procesamiento para una GPU



Fuente: Elaboración propia

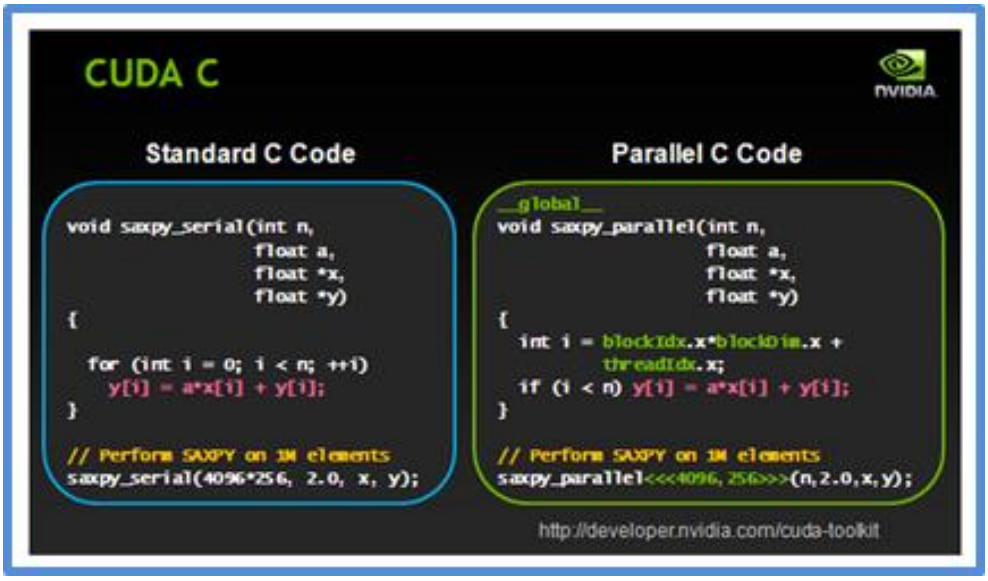
Después de que la GPU recibe los datos vertex (vértices) desde el host (CPU), la fase vertex se ejecuta en primer lugar. La función de fijado transforma la imagen y el hardware de luminosidad operado en esta fase se lleva a cabo; entonces los píxeles shaders programables, y el control de flujo dinámico de los modelos shaders entran en juego. El siguiente paso en el pipeline clásico es la configuración, donde los vértices son ensamblados dentro de primitivas como triángulos, líneas o puntos. Las primitivas son convertidas por la fase de "rasterización" en fragmentos de píxeles (o simplemente fragmentos), pero no son considerados píxeles completos en esta fase. Los fragmentos están sometidos a muchas otras operaciones. Los fragmentos son finalmente considerados píxeles cuando han sido escritos en el buffer frame. (Sánchez, 2008).

A continuación, la siguiente fase es la de píxel shader, En el pasado, los fragmentos sólo podían haber tenido valores de color aplicados de textura simple. Hoy en día, la capacidad de sombreado de un píxel programado de la GPU permite numerosos efectos de sombreado para ser aplicados mientras se trabaja de acuerdo con métodos complejos de multitextura. Específicamente, los fragmentos sombreados (con color y valores Z) desde esta fase píxel son enviados al ROP (Raster Operations). La fase ROP es donde se chequea el buffer Z para asegurar que sólo los fragmentos visibles son procesados rápidamente, y los fragmentos visibles, si son parcialmente transparentes, son mezclados con el buffer de frame existente, junto con los píxeles. El píxel procesado final es enviado a la memoria buffer para ser escaneado y visualizado en el monitor. (NVIDIA CUDA, 2006)

4.2.2.3. Procesamiento en paralelo

Más específicamente, la GPU está especialmente pensada para direccionar problemas que pueden ser expresados como computaciones de datos paralelos (el mismo programa es ejecutado en muchos elementos de datos en paralelo) con gran intensidad aritmética (el ratio de operaciones aritméticas respecto a operaciones de memoria). Como el mismo programa es ejecutado para cada elemento de datos, hay menos requisitos para un flujo de control sofisticado; y como es ejecutado en muchos elementos de datos y tiene gran intensidad aritmética, la latencia de acceso a memoria puede ser ocultada con cálculos, en vez de datos muy grandes de caché. (NVIDIA CUDA, 2008)

Figura 19: Diferencias entre codificación en paralelo y serie



Fuente: (NVIDIA Corporation, 2015)

El procesamiento de datos paralelos acota los elementos de datos al procesamiento paralelo de hilos. Muchas aplicaciones que procesan grandes conjuntos de datos como vectores, pueden usar un modelo de programación de datos paralelos para acelerar los cálculos. En renderizado 3D los conjuntos de píxeles y vértices se asignan a hilos paralelos. De la misma manera, aplicaciones de procesamiento de imágenes y media como post-procesado de imágenes renderizadas, codificación y decodificación de video, escalado de imágenes, visión estéreo, y patrones de reconocimiento pueden asociar bloques de la imagen y píxeles a hilos de procesamiento paralelo. De hecho, muchos algoritmos fuera del campo del renderizado como el procesamiento de señales, simulaciones físicas finanzas o biología, se aceleran con el procesamiento de datos en paralelo.

Hasta la fecha, sin embargo, a pesar de acceder a todo el poder de computación contenido en al GPU y usarlo eficientemente para aplicaciones científicas, seguía siendo difícil obtener las siguientes pautas:

- La GPU solamente podía ser programada a través de la API (Application Programming Interface) gráfica; esto provocaba que la curva de aprendizaje

para un desarrollador principiante fuese muy elevada, ya que tenía que trabajar con una API inadecuada, que no estaba adaptada a la aplicación científica.

- La DRAM de la GPU podía ser leída de manera general (los programas de GPU pueden obtener elementos de datos de cualquier parte de la DRAM) pero no se podía escribir de manera general (los programas de GPU no pueden esparcir la información a cualquier parte de la DRAM), eliminando mucha de la flexibilidad de programación ya disponible en la CPU.
- Algunas aplicaciones tenían en problema del “cuello de botella”, debido al ancho de banda de la memoria DRAM, utilizando escasamente el poder computacional de la GPU.

4.2.3. CUDA y cálculo de GPU

CUDA viene del inglés Compute Unified Device Architecture y es una nueva arquitectura hardware y software, diseñada para dar y manejar procesamientos en la GPU como un elemento de computación de datos paralelos sin la necesidad de mapearlos a una API de gráficos. Está disponible para las versiones GeForce 8 Series, Quadro FX 5600/4600, y Tesla. El mecanismo de multitarea del sistema operativo es responsable de manejar el acceso a la GPU mediante CUDA, y las aplicaciones gráficas funcionan de forma simultánea. A continuación describimos el pipeline unificado del que disponen las actuales GPUs de NVIDIA y que puede ser explotado de forma eficiente mediante CUDA. (Sánchez, 2008).

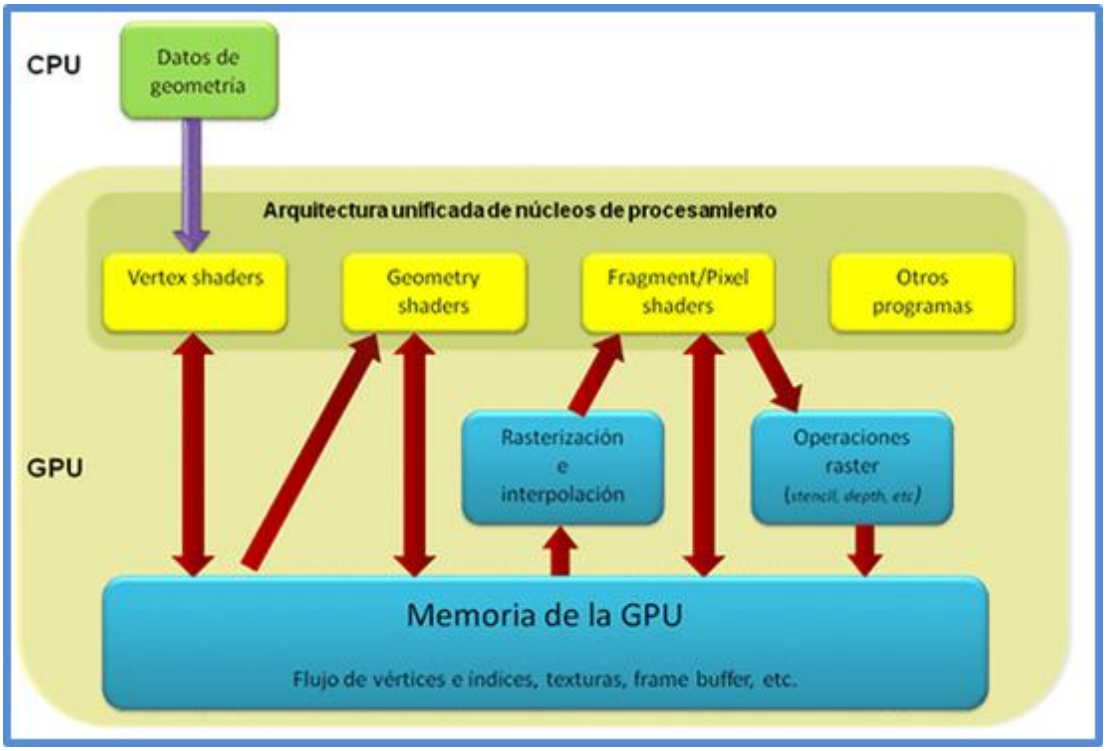
4.2.3.1. Pipeline unificado

A partir del modelo de pipeline clásico, con sus flujos de datos empezando en lo más alto, donde los vértices con varios atributos, índices, comandos, y texturas son pasados a la GPU desde la CPU. Las fases de procesamiento mayores

siguen una manera lineal segura incluyendo vertex shading, píxel shading, operaciones raster[25] y escritura de píxeles en el buffer frame.

Con este pipeline unificado y la arquitectura “shader”, el diseño de la GPU reduce significativamente el número de fases del pipeline y cambia el flujo secuencial para estar más orientado a bucle. Las entradas son alimentadas en la parte alta del núcleo shader unificado, y las salidas son escritas en registros y entonces vuelven otra vez a la parte alta del núcleo shader para la próxima operación. Como resultado, en el diagrama GPU unificado generalizado que se muestra en la figura, los flujos de datos bajan secuencialmente por el pipeline a través de diferentes tipos “shader”. (NVIDIA CUDA, 2006)

Figura 20: Pipeline unificado



Fuente: (Charte , 2009)

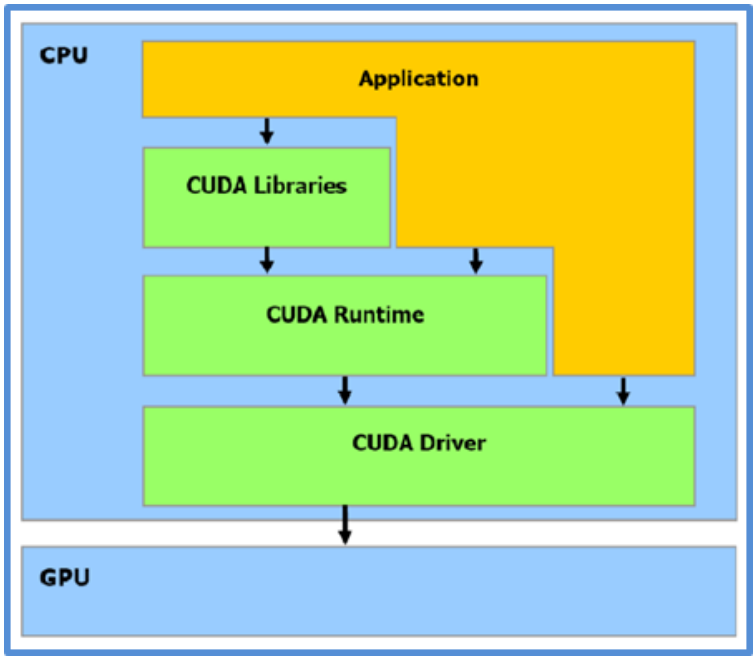
[25] Son operaciones a través de las cuales un área espacial queda dividida en celdas regulares, en las que cada una de las cuales presentan unos atributos o valor, como pueden ser la altitud, reflectancia, etc.

Como puede apreciarse en anterior figura, los datos vienen de la parte superior izquierda del diseño unificado (como vértices), y son llevados al núcleo shader para su procesamiento, y los resultados son enviados de vuelta a la parte superior del núcleo shader, donde son llevados otra vez, procesados otra vez, mandados de vuelta a la parte superior, y así hasta que todas las operaciones shader son ejecutadas y el fragmento de píxel se pasa al subsistema ROP. (Sánchez, 2008)

4.2.3.2. Programación en CUDA

Antes de profundizar en el modelo de programación empleado por CUDA, destacamos que la pila del software de CUDA se compone de varias capas, tal y como muestra en el diagrama de bloques. En concreto, dichas capas son un controlador de hardware, una API y su runtime, y dos librerías matemáticas de alto nivel para uso común, CUFFT y CUBLAS. El hardware ha sido diseñado para soportar controladores ligeros y capas runtime, dando como resultado una ejecución óptima. En este sentido, la API de CUDA es una extensión del lenguaje de programación C, lo cual hace que tenga una curva de aprendizaje mínima.

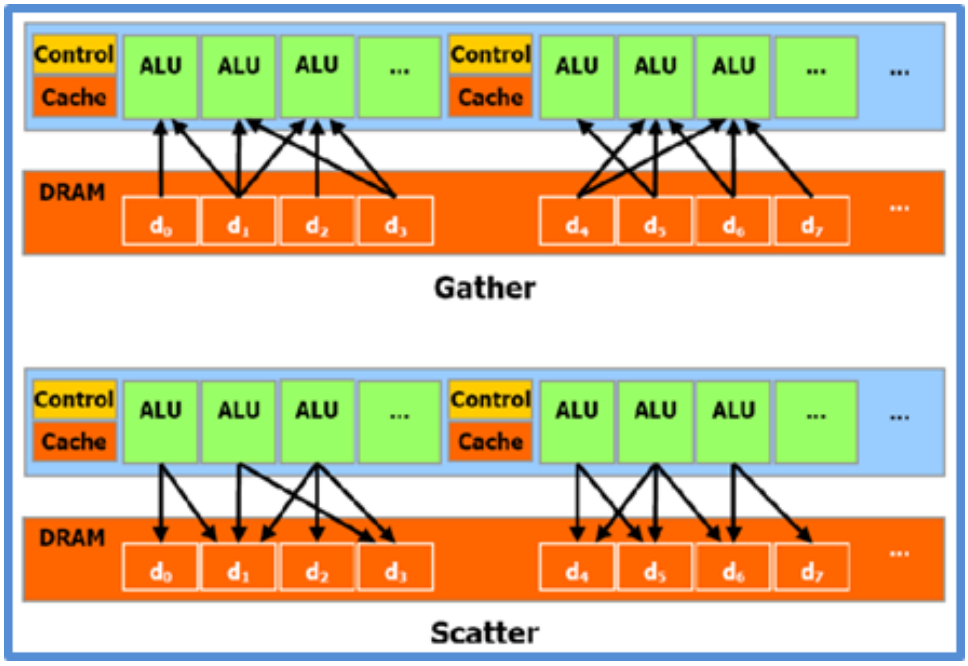
Figura 21: Diagrama de bloques de CUDA



Fuente: (Sánchez, 2008)

Por otra parte, CUDA ofrece un direccionamiento de carácter general para la memoria DRAM. Este modelo de direccionamiento permite obtener mayor flexibilidad en la programación, en el sentido de que ofrece tanto la operación de reparto de datos como la de obtención de estos. Desde una perspectiva de programación, esto se traduce en la habilidad de leer y escribir datos en cualquier lugar de la DRAM, exactamente igual que en la CPU (NVIDIA CUDA, 2007)

Figura 22: Operaciones de memoria gather (dispersión) y scatter (reunión)



Fuente: (Sánchez, 2008)

CAPÍTULO V

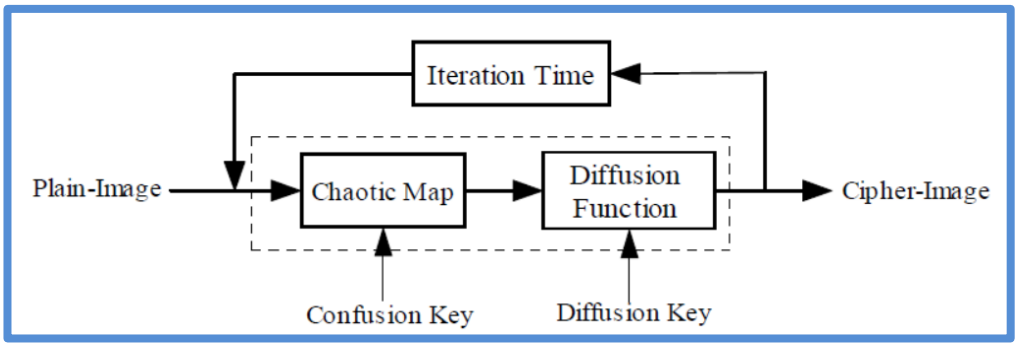
MÉTODOS

5.1. IMPLEMENTACIÓN

El objetivo de cualquier sistema de criptografía es convertir un texto en plano a un texto cifrado a través del uso de un algoritmo seguro, La implementación de la solución está basada en los conceptos expuestos en el punto 3.3.4 relacionados con proceso de confusión y difusión aplicadas a las imágenes digitales. En este sentido se explica la arquitectura base sobre la cual se implementa el algoritmo.

El sistema de criptografía de imágenes basado en caos consiste principalmente en dos etapas en la que una imagen en plano se constituye en el "input" del sistema. A continuación se ejecuta el proceso de confusión en la que se aplica un mapa caótico, para el presente trabajo se aplica el mapa logístico. Posteriormente, el proceso de difusión entra en escena dentro la implementación para fortificar aún más la seguridad. Ambos procesos principales requieren de una clave secreta que, en el ámbito de seguridad de sistemas, viene a ser la llave secreta. Aunque ambos procesos ejecutados en forma serial robustecen la seguridad, iterar un número determinado de veces incrementa aún más la confiabilidad del sistema criptográfico. Finalmente, al culminar la última iteración se obtiene el "output" que es denominada imagen cifrada de la cual a simple vista no representa la imagen original. Para una mejor comprensión de esta arquitectura se presenta la siguiente figura:

Figura 23: Esquema de implementación



Fuente: (Lian, Sun, & Wang, 2005)

5.2. ALGORITMOS

Con la intención de elevar la sensibilidad en las condiciones iniciales ante ruidos externos para que una aproximación en los parámetros por parte del hacker no cause una imagen parcialmente descifrada, se aplica conceptos de teoría del caos en los procesos principales descritos en la implementación. Es decir, se utiliza el mapa logístico tanto para el procedimiento de confusión, en primera instancia; y seguidamente en el procedimiento de difusión.

5.2.1. Algoritmo de Encriptación

La comunicación segura consta en una primera acción denominada la encriptación en la que se busca cifrar el mensaje a través de un algoritmo. En el presente trabajo se busca cifrar una imagen digital a través de un algoritmo de encriptación que utilice conceptos de teoría del caos. El resulta de este proceso será una imagen que por sí misma no represente ningún mensaje. A continuación el detalle de la encriptación.

5.2.1.1. Proceso de Confusión

De acuerdo a la recopilación bibliográfica, el proceso de confusión que hace uso de una ecuación logística dentro sus cálculos es el presentado en (Jiménez Rodríguez, Jaimes Reategui, & N. Pisarchik, 2012). En este entendido, el procedimiento de confusión viene descrito a continuación:

Paso 1: Si bien la imagen digital es representada por una matriz, el primer paso es trasformar dicha matriz en un vector lineal cuyos elementos son los valores de pixeles de la imagen, La matriz debe ser recorrida empezando por la primera columna de izquierda a derecha; y posteriormente por filas de arriba hacia abajo. El tamaño del vector viene dado por las dimensiones de la imagen, es decir, una imagen digital de $M \times N$ pixeles es trasformada en un vector de $h = MN$ elementos. Por tanto, el vector será:

$$PT = [P_1, P_2, P_3, \dots, P_{MN}]$$

Donde P_i representa el valor del pixel de la imagen digital.

Paso 2: Convertir el vector de la imagen en plano PT de valores enteros en un vector de valores decimales dentro del intervalo $[0,1]$. De esta manera se sitúan dentro de un atractor caótico del mapa logístico. Este nuevo vector lo denominaremos de la siguiente manera:

$$PN = \frac{PT}{255} = [p_1, p_2, p_3, \dots, p_i, \dots, p_h]$$

El valor de 255 viene dado por la cantidad de colores que puede contener una imagen en escala de grises esencialmente.

Paso 3: En este paso, se aplica la ecuación logística descrita en el punto 3.2.5 que viene definida por:

$$x_{k+1} = \mu x_k (1 - x_k)$$

En la que la llave secreta está conformada por el valor de la constante μ y el valor de la condición inicial x_0. Con estos parámetros definidos, se debe iterar h veces la ecuación logística para obtener la secuencia caótica representa por el siguiente vector:

$$LM = [x_1, x_2, x_3, \dots, x_n, \dots, x_h]$$

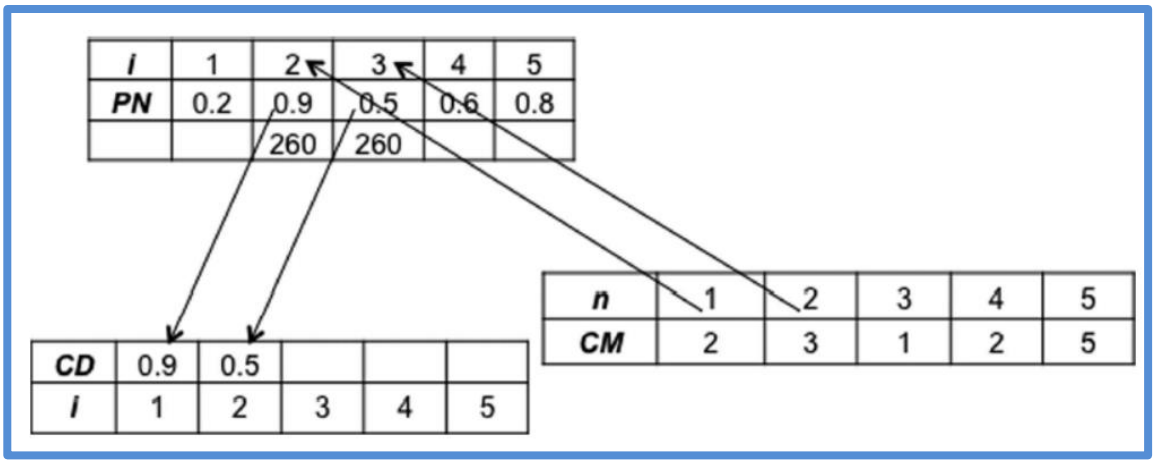
Cabe aclarar que el tamaño del vector LM tiene la misma longitud h que el vector PN.

Paso 4: En este paso se define las nuevas posiciones que adoptarán cada pixel dentro de la imagen digital. En este sentido se trasforma el vector de decimales LM en un vector de enteros CM cuyo rango de valores oscila en el intervalo $[1, h]$. De esta manera, se obtiene la clave caótica de mezcla que indica las nuevas posiciones de los pixeles en la nueva imagen normalizada en plano a través del siguiente cálculo:

$$CM = round[LM(h-1)] + 1 = [k_1, k_2, \dots, k_n, \dots, k_h]$$

Paso 5: Para la lograr la confusión se hace uso de los vectores obtenidos en los pasos precedentes. El proceso se inicia tomando uno a uno los elementos k_n del vector CM y utilizarlo como indicador de la nueva posición que adoptará el elemento correspondiente del vector PN para crear el vector resultante que se denomina CD. Este proceso debe ser realizado iniciando por el primer componente del vector CD y continuar sucesivamente hasta finalizar con el último de ellos como se muestra en la siguiente figura:

Figura 24: Ilustración del proceso de Confusión



Fuente: (Jiménez Rodríguez, Jaimes Reategui, & N. Pisarchik, 2012)

El valor 260 debajo de algunos elementos del vector PN tiene la función de actuar como bandera que indique que el elemento ya fue situado en su nueva posición y de esa manera si el uno de los elemento del vector CM apunta por segunda vez al mismo elemento procesado, la bandera indicará que debe seleccionarse otro elemento y de esta manera se evita el problema de posicionar doblemente un mismo elemento.

Cuando finalice el recorrido del vector CM, el vector CD debe tener todos los elementos completados. Es en este punto donde finaliza el proceso de confusión

5.2.1.2. Proceso de Difusión

Continuando con la base descrita en la arquitectura de implementación del sistema criptográfico, el valor de los pixeles es modificado a través de una secuencia generada por un segundo sistema caótico que acompaña al proceso de confusión. A continuación se explica los pasos del proceso de difusión:

Paso 6: El proceso de difusión inicia aplicando la ecuación logística para alcanzar un comportamiento caótico a través de la ecuación:

$$x_{k+1} = \mu x_k(1 - x_k)$$

Como se ha explicado en la sección teórica, el valor de la constante μ y el valor de la condición inicial x_0 son parámetros clave que en conjunto conforman la llave secreta la cual debe ser conocida por el emisor y receptor para que la comunicación sea correcta en el envío del mensaje, en este caso una imagen digital. Por tanto, en este paso se definen el valor inicial x_0, la constante de la ecuación logística μ y finalmente el tamaño del vector dado por $h = M \times N$, donde M representa el número de pixeles por columna de la imagen y N el número de pixeles por fila.

Paso 7: Se genera el vector caótico en base a la ecuación logística que es iterada h veces y cuyos valores se almacenan en una cola. Cada nuevo valor es el resultado de la ecuación logística que utiliza entre sus parámetros el valor de la iteración anterior. En general, los valores se tienen:

$$X = [x_1, x_2, x_3, \dots, x_h]$$

Paso 8: A continuación se genera en base al anterior vector, un nuevo vector de números enteros que oscilen en el intervalo $[1{,}255]$ a través de la fórmula:

$$y_n = mod[E(1000x_n), 256]$$

En la cual se debe obtener la parte entera de la multiplicación de 1000 por el ítem x_i que corresponda y posteriormente hallar el módulo de dicho resultado entre 256. El resultado final debe almacenarse en el nuevo vector. Dicho vector Y tiene el mismo tamaño que el vector X, es decir, la cantidad de pixeles de la imagen a cifrar, por tanto se tiene:

$$Y = [y_1, y_2, y_3, \dots, y_h]$$

Paso 9: Generar un vector unidimensional I que representa el valor de todos los pixeles de la imagen (en escala de grises), quedando la representación genérica de la siguiente manera:

$$I = [i_1, i_2, i_3, \dots, i_{M \times N}]$$

Se comienza en la esquina superior izquierda y se desplaza de izquierda a derecha y de arriba hacia abajo.

Paso 10: Este paso es el principal dentro del proceso de difusión ya que recorre la imagen que sirve de entrada y cambia su valor. Para tal efecto se hace uso tanto del vector Y como del vector I. Debido a que ambos tienen la misma dimensión y cantidad de elementos, se aplica la operación XOR de módulo 2 entre ambos vectores emparejados por su índice, es decir:

$$i_n^* = y_n \oplus i_n \qquad \text{para } 1 \leq n \leq h$$

Para realizar la operación XOR, se debe transformar los sumandos en sus equivalencias binarias y aplicar la tabla de verdad correspondiente:

Tabla 6: Tabla de Verdad de la operación XOR

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

Fuente: Elaboración propia

El resultado en binario obtenido debe ser transformado nuevamente a su equivalencia decimal constituyendo el valor definitivo del pixel de la imagen cifrada resultante. El proceso de difusión descrito fue basado en la propuesta de (Rojas & Cano, 2011).

5.2.2. Algoritmo de Desencriptación

Culminado el envío seguro de la imagen digital, el receptor requiere de un algoritmo que le permita descifrar el mensaje y obtener, en este caso, la imagen original. No tendría sentido ni lógica que el envío sea seguro y protegido contra agentes externos, pero que al mismo tiempo no sea disponible para los destinatarios correctos. A continuación se detalla el proceso de desencriptación que al igual que la encriptación, se rige bajo la arquitectura de dos etapas, primero difusión y finalmente confusión.

5.2.2.1. Proceso de Difusión

Para iniciar la desencriptación, el primer proceso es la difusión en la que se tiene como objetivo recuperar los valores originales de los pixeles que oscilan entre $[1,255]$ y representan el grado de intensidad de las imagen en escala de grises. Al igual que la encriptación, se lo realiza a través de una secuencia generada por el

primer sistema caótico. A continuación se explica los pasos del proceso de difusión:

Paso 1: El proceso de difusión inicia aplicando la ecuación logística para alcanzar un comportamiento caótico con el cual fue encriptado a través de la ecuación:

$$x_{k+1} = \mu x_k (1 - x_k)$$

El valor de la constante μ y el valor de la condición inicial x_0 son parámetros clave que el receptor debe tener conocimiento antes de ejecutar la aplicación para que la comunicación sea correcta en el envío del mensaje, en este caso una imagen digital. Por tanto, en este paso se definen el valor inicial x_0, la constante de la ecuación logística μ y finalmente el tamaño del vector dado por $h = M \times N$, donde M representa el número de pixeles por columna de la imagen cifrada y N el número de pixeles por fila.

Paso 2: Se genera el vector caótico en base a la ecuación logística que es iterada h veces y cuyos valores se almacenan en una cola. Debido a que se hace uso de los mismos parámetros usados en la encriptación, el resultado debiera ser el mismo. Cada nuevo valor es el resultado de la ecuación logística que utiliza entre sus parámetros el valor de la iteración anterior. En general, los valores se tienen:

$$X = [x_1, x_2, x_3, \dots, x_h]$$

Paso 3: Al igual que en la encriptación, se genera en base al anterior vector, un nuevo vector de números enteros que oscilen en el intervalo $[1{,}255]$ a través de la fórmula:

$$y_n = mod[E(1000x_n), 256]$$

En la cual se debe obtener la parte entera de la multiplicación de 1000 por el ítem x_i que corresponda y posteriormente hallar el módulo de dicho resultado entre 256. El resultado final debe almacenarse en el nuevo vector. Dicho vector Y tiene

el mismo tamaño que el vector X, es decir, la cantidad de pixeles de la imagen cifrada, por tanto se tiene:

$$Y = [y_1, y_2, y_3, \dots, y_h]$$

Paso 4: Generar un vector unidimensional I que representa el valor de todos los pixeles de la imagen cifrada (en escala de grises), quedando la representación genérica de la siguiente manera:

$$I = [i_1, i_2, i_3, \dots, i_{M \times N}]$$

Se comienza en la esquina superior izquierda y se desplaza de izquierda a derecha y de arriba hacia abajo.

Paso 5: La imagen cifrada que sirve de entrada, recupera su valor original. Para tal efecto se hace uso tanto del vector Y como del vector I. Debido a que ambos tienen la misma dimensión y cantidad de elementos, se aplica la operación XOR de módulo 2 entre ambos vectores emparejados por su índice, es decir:

$$i_n^* = y_n \oplus i_n \qquad \text{para } 1 \leq n \leq h$$

Para realizar la operación XOR, se debe transformar los sumandos en sus equivalencias binarias y aplicar la misma tabla de verdad utilizada en la encriptación. De esta manera se recuperan los valores originales de los pixeles aunque aún se encuentran en desorden. Por lo tanto, para hallar el orden correcto, se aplica el proceso de confusión en su forma inversa como se explica a continuación.

5.2.2.2. Proceso de Confusión

Después de recuperar los valores reales de los pixeles, existe la necesidad de ordenar y asignar las posiciones correctas en el plano de imagen. Para este propósito se realizan los siguientes pasos:

Paso 6: La imagen cifrada se encuentra representada por una matriz, el primer paso es trasformar dicha matriz en un vector lineal cuyos elementos son los

valores de pixeles de la imagen encriptada, La matriz debe ser recorrida empezando por la primera columna de izquierda a derecha; y posteriormente por filas de arriba hacia abajo. El tamaño del vector viene dado por las dimensiones de la imagen cifrada, es decir, una imagen digital de $M \times N$ pixeles es trasformada en un vector de $h = MN$ elementos. Por tanto, el vector será:

$$PT = [P_1, P_2, P_3, \dots, P_{MN}]$$

Donde P_i representa el valor del pixel de la imagen digital.

Paso 7: Convertir el vector de la imagen cifrada PT de valores enteros en un vector de valores decimales dentro del intervalo $[0,1]$. De esta manera se sitúan dentro de un atractor caótico del mapa logístico. Este nuevo vector lo denominaremos de la siguiente manera:

$$PN = \frac{PT}{255} = [p_1, p_2, p_3, \dots, p_i, \dots, p_h]$$

El valor de 255 viene dado por la cantidad de colores que puede contener una imagen en escala de grises esencialmente.

Paso 8: En este paso, se aplica la ecuación logística descrita en el punto 3.2.5 que viene definida por:

$$x_{k+1} = \mu x_k (1 - x_k)$$

En este paso, el receptor debe utilizar la clave secreta que está conformada por el valor de la constante μ y el valor de la condición inicial x_0. Con estos parámetros definidos, se debe iterar h veces la ecuación logística para obtener la secuencia caótica representa por el siguiente vector:

$$LM = [x_1, x_2, x_3, \dots, x_n, \dots, x_h]$$

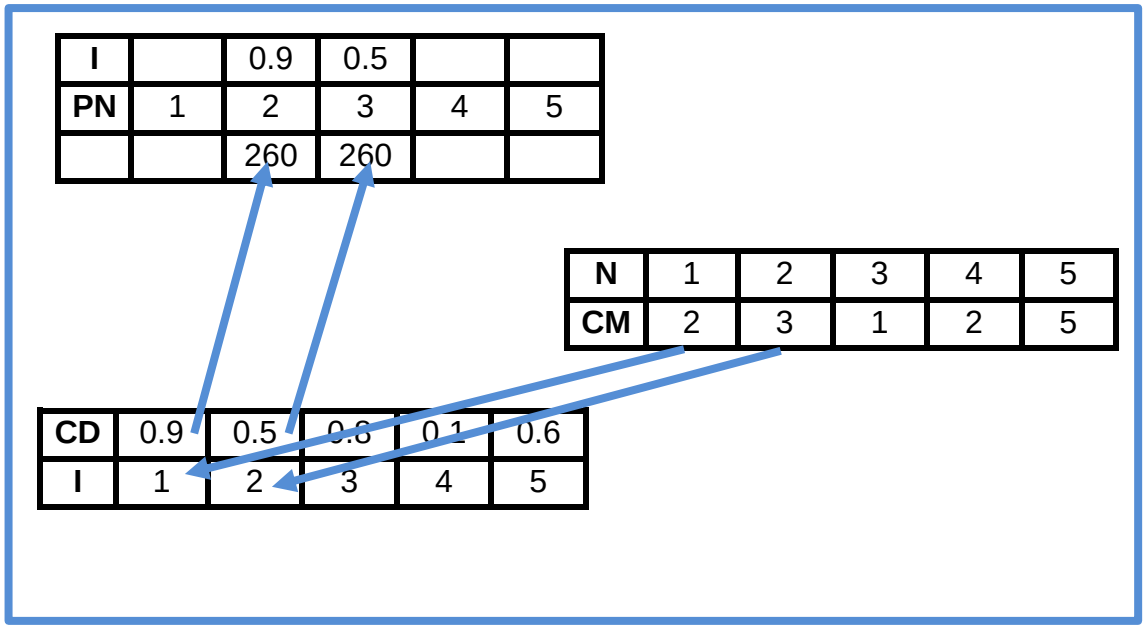
Cabe aclarar que el tamaño del vector LM tiene la misma longitud h que el vector PN.

Paso 4: En este paso se ordena y asignan posiciones originales que adoptarán cada pixel de la imagen cifrada para obtener la imagen original. En este sentido se trasforma el vector de decimales LM en un vector de enteros CM cuyo rango de valores oscila en el intervalo $[1, h]$. De esta manera, se obtiene la clave caótica de mezcla que indica las posiciones desde las cuales se debe ordenar los pixeles. Debido a que se utiliza la misma llave de la encriptación, el vector CM debe contener los mismos elementos que se generaron en la encriptación.

$$CM = round[LM(h-1)] + 1 = [k_1, k_2, \dots, k_n, \dots, k_h]$$

Paso 5: Para ordenar los pixeles a sus posiciones correctas, se hace uso de los vectores obtenidos en los pasos precedentes. El proceso se inicia tomando uno a uno los elementos k_n del vector CM y utilizarlo como indicador de la posición original que adoptará el elemento correspondiente del vector CD para crear el vector resultante que se denomina PN. Este proceso debe ser realizado iniciando por el primer componente del vector CD y continuar sucesivamente hasta finalizar con el último de ellos como se muestra en la siguiente figura:

Figura 25: Ilustración del proceso de Confusión para la desencriptación



Fuente: Elaboración propia

El valor 260 debajo de algunos elementos del vector PN tiene la función de actuar como bandera que indique que el elemento ya fue situado en su posición original y de esa manera si uno de los elementos del vector CM apunta por segunda vez al mismo elemento procesado, la bandera indicará que debe seleccionarse otro elemento y de esta manera se evita el problema de posicionar doblemente un mismo elemento.

Cuando finalice el recorrido del vector CM, el vector PN debe tener todos los elementos completados. Es en este punto donde finaliza el proceso de confusión para la desencriptación.

5.3. INNOVACION

El presente trabajo está basado en una recopilación de varias propuestas entre las cuales se destaca la arquitectura presentada en (Pisarchik & Zanin, 2008) respecto a las etapas de confusión y difusión. En dicha propuesta, se utiliza dos parámetros α y β que tienen la función de contar el número de iteraciones de cada etapa respectivamente. Aunque este hecho pueda incrementar la robustez del algoritmo, se considera que un solo parámetro variable alcanza en iguales magnitudes la fortaleza del sistema criptográfico. A diferencia de propuesta descrita, el número de iteraciones no se define como un parámetro del usuario, sino que utiliza el número de pixeles de la imagen a cifrar. Esta innovación otorga al algoritmo versatilidad y dinámica para una determinada imagen digital.

Por otro lado, otra mejora a las propuestas descritas en los antecedentes es la combinación de algoritmos utilizados en cada etapa. Para la etapa de confusión se aplica la primera mitad del algoritmo presentado en (Jiménez Rodríguez, Jaimes Reategui, & N. Pisarchik, 2012) a través del uso de la primera ecuación logística. Por otro lado, para la etapa de difusión, se aplica el algoritmo propuesto en (Rojas & Cano, 2011) a través del uso de la segunda ecuación logística. Mientras otras propuestas exponen mapas caóticos de dos dimensiones para mejorar la

seguridad, el presente trabajo propone la combinación de dos mapas caóticos unidimensionales pero relacionados de manera serial.

Finalmente, tecnología en pleno crecimiento y aplicación como lo son los procesadores de imágenes y video forman parte de la originalidad del presente trabajo. Tarjetas de video NVIDIA se ha encargado de agilizar el procesamiento de imágenes a través de la unidad GPU y estos avances fueron aprovechados en el sistema criptográfico expuesto en este trabajo. La estrecha relación de las librerías que acceden a este tipo de memoria con OpenCV permite reducir tiempos de procesamiento. Debido a que mejorar la eficiencia forma parte de los objetivos iniciales, el algoritmo diseñado está programado bajo el lenguaje C++ de bajo nivel y acompañado de las librerías que acceden no solo a la memoria CPU, sino a la memoria propia de tarjetas gráficas que permiten ejecutar procesos paralelos.

CAPÍTULO VI

PRUEBAS Y VALIDACIÓN

6.1. PRUEBAS DE FUNCIONAMIENTO

Estas pruebas tienen la intensión de comprobar que el proceso tanto de encriptación como desencriptación se ejecutan correctamente con imágenes en escala de grises las cuales se encuentran dentro el alcance del presente trabajo. Por lo tanto, se utiliza una imagen clásica en este tipo de pruebas que manipula imágenes digitales a nivel de pixeles, la imagen es la siguiente:

Figura 26: Imagen en escala de grises para pruebas de funcionamiento "cam_74.pgm"

A continuación ejecutamos el compilado "Encriptacion.exe" en la que se ingresan los siguientes parámetros:

[1] Nombre de la imagen que se desea encriptar

[2] Nombre del archivo resultante (imagen cifrada) con extensión .png

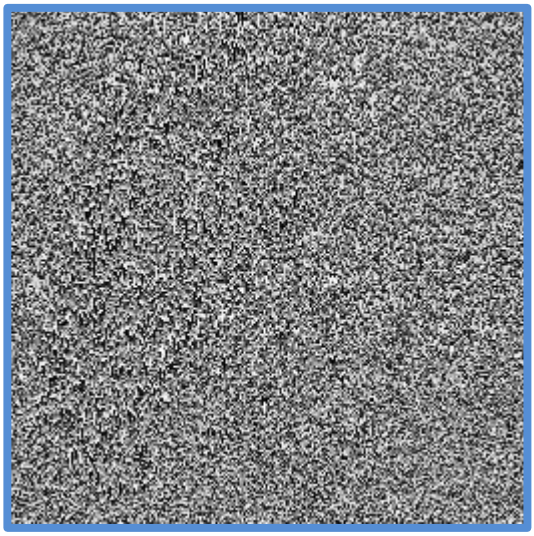
[3] Valor en decimal que corresponda a la condición inicial x_0

En este caso el comando a ejecutar para cada sistema operativo son:

- Para Windows: Encriptacion.exe cam_74.pgm out.png 0.6530
- Para Linux: ./Encriptacion cam_74.pgm out.png 0.6530

El resultado obtenido, denominado en este caso “out.png” es el siguiente:

Figura 27: Imagen cifrada “out.png”



Fuente: Elaboración propia

De esta manera se comprueba que el algoritmo no presenta error de código y que ha completado satisfactoriamente el proceso de encriptación. A continuación se debe realizar la prueba de desencriptación para verificar que se puede recuperar la imagen original sin problemas en la ejecución.

Al igual que el proceso de encriptación, para la desencriptación se ejecuta el archivo “Desencriptacion.exe” que tiene la misma estructura de parámetros, es decir:

[1] Nombre de la imagen que se desea desencriptar

[2] Nombre del archivo resultante (imagen descifrada) con extensión .png

[3] Valor en decimal que corresponda a la condición inicial x_0

En este caso el comando a ejecutar para cada sistema operativo son:

- Para Windows: Desencriptacion.exe out.png original.png 0.6530
- Para Linux: ./Desencriptación out.png original.png 0.6530

El resultado obtenido, denominado en este caso “original.png” es el siguiente:

Figura 28: Imagen descifrada “original.png”

Fuente: Elaboración propia

Como se puede observar, la imagen original es recuperada de manera exitosa después de ejecutar el algoritmo para descifrar lo que asegura el correcto funcionamiento tanto de la encriptación como el proceso inverso en imágenes digitales. El siguiente tipo de pruebas pretende demostrar principalmente el efecto que produce el comportamiento caótico y la sensibilidad en las condiciones iniciales.

6.2. PRUEBAS DE COMPORTAMIENTO CAÓTICO

Como bien se ha explicado en los anteriores capítulos, la teoría del caos y más precisamente el mapa logístico presenta una serie de propiedades de gran ayuda para la generación de números pseudo-aleatorios.

En primer lugar se debe tener en cuenta que la periodicidad es muy importante ya que para proteger la imagen digital, el proceso de difusión que se encarga de cambiar el valor de los pixeles de manera aleatoria no debe converger en un mismo valor ya que facilitaría la labor de terceras personas. Para demostrar esta propiedad se ha generado 100 iteraciones de la ecuación logística partiendo de la

condición inicial $x_0 = 0.1$ y el resultado de las últimas iteraciones para distintas constantes μ es la siguiente:

Para $\mu = 2.7 \rightarrow \{\dots, 0.6296, 0.6296, 0.6296, 0.6296, 0.6296, 0.6296, 0.6296, 0.6296\}$

Para $\mu = 3.3 \rightarrow \{\dots, 0.4794, 0.8236, 0.4794, 0.8236, 0.4794, 0.8236, 0.4794\}$

Como se observa, el valor que se otorgue a la contante μ es crucial y por tanto debe ser controlado por el mismo algoritmo. En el primer caso observado, la sucesión converge en el punto 0.6296 y en el segundo caso se observa un ciclo de periodo 2. Ambos ejemplos resultan fáciles de descifrar para personas ajenas. Sin embargo, si se aplica el valor $\mu = 4$ y partiendo de la misma condición inicial $x_0 = 0.1$, no existe ninguna convergencia en la sucesión por mas iteraciones que se realice. De hecho el rango que se puede utilizar es $3.5699456 < \mu \leq 4$.

En segundo lugar, la sensibilidad a las condiciones iniciales es otra propiedad importante de la cual se toma partido en la encriptación de imágenes digitales. Para demostrar esta propiedad se realiza la prueba generando tres secuencias caóticas que bien podrían utilizarse en los procesos de confusión y difusión. Para una constante no caótica $\mu = 2.7$ y condiciones iniciales distintas, se realizan 100 iteraciones y obtenemos:

Para $x_0 = 0.1 \rightarrow \{\dots, 0.6296, 0.6296, 0.6296, 0.6296, 0.6296, 0.6296, 0.6296\}$

Para $x_0 = 0.101 \rightarrow \{\dots, 0.6296, 0.6296, 0.6296, 0.6296, 0.6296, 0.6296, 0.6296\}$

Para $x_0 = 0.9 \rightarrow \{\dots, 0.6296, 0.6296, 0.6296, 0.6296, 0.6296, 0.6296, 0.6296\}$

La interpretación de estos resultados indican que sin importar la condición inicial que se utilice en el algoritmo, si la constante caótica es inferior a 3.56 no representará el caos que se busca y de la cual se pretende aprovechar. Es decir, sea cual sea el valor de x_0 de inicio de la iteración, es convergente al punto 0.6296. Sin embargo, si $\mu = 4$ se obtiene sucesiones totalmente distintas incluso si los valores iniciales son muy cercanos entre ellos:

Para $x_0 = 0.1 \rightarrow \{\ldots, 0.4795, 0.9983, 0.0067, 0.0267, 0.1039, 0.3724\}$

Para $x_0 = 0.101 \rightarrow \{\ldots, 0.8371, 0.5455, 0.9917, 0.0329, 0.1271, 0.4439\}$

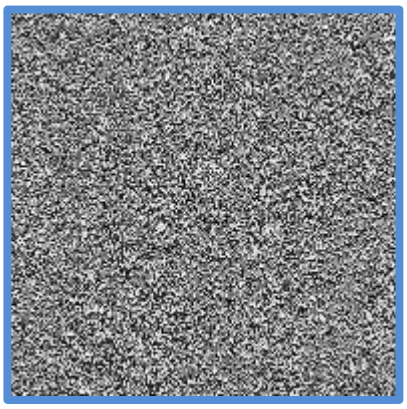
Para una demostración más visual, se encriptará otra imagen clásica de pruebas con condición inicial $x_0 = 0.6530$, es decir, el comando será:

- Para Windows: Encriptacion.exe lena.png out.png 0.6530
- Para Linux: ./Encriptacion lena.png out.png 0.6530

Figura 29: Imagen en escala de grises para pruebas de funcionamiento “lena.png”

El resultado obtenido, denominado en este caso “out.png” es el siguiente:

Figura 30: Imagen descifrada “out.png”



Fuente: Elaboración propia

Ya que se tiene la imagen encriptada de la cual a simple vista no se puede observar el contenido real, se desencripta de dos maneras, La primera utilizando

la misma llave con la cual fue encriptada y la segunda con una llave muy cercana al valor utilizado para verificar la sensibilidad a las condiciones iniciales. Por tanto la primera ejecución es:

- Para Windows: Desencriptacion.exe out.png original_Lena.png 0.6530
- Para Linux: ./Desencriptación out.png original_Lena.png 0.6530

Y el resultado es efectivamente el esperado, la imagen original descifrada porque se ha utilizado la misma clave.

Figura 31: Imagen descifrada “original_Lena.png” con la llave correcta

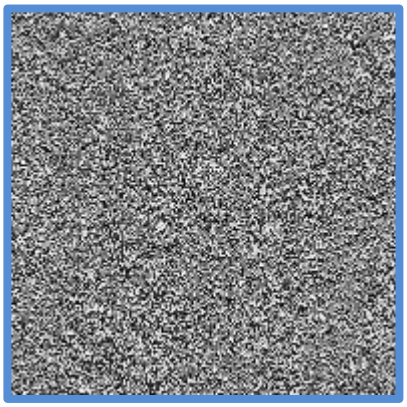
Fuente: Elaboración propia

A continuación se realiza la desencriptación, pero esta vez utilizando el valor de la condición inicial muy cercana a la clave correcta para ver si recupera la imagen original o al menos una cercana, el comando es:

- Para Windows: Desencriptacion.exe out.png original_Lena.png 0.6531
- Para Linux: ./Desencriptación out.png original_Lena.png 0.6531

Y el resultado demuestra que el algoritmo es sensible a las condiciones iniciales y por un mínimo cambio en las condiciones iniciales que son la llave del sistema criptográfico, no es posible visualizar la imagen:

Figura 32: Imagen descifrada “original_Lena.png” con la llave incorrecta



Fuente: Elaboración propia

6.3. PRUEBAS DE EFICIENCIA

Finalmente, se realizan las pruebas de eficiencia en la que se tiene por objeto medir el tiempo de procesamiento. Debido la arquitectura describe un algoritmo dinámico dependiendo del tamaño de la imagen digital, se realizan pruebas sobre imágenes de distinto tamaño utilizando por una parte el procesamiento CPU y por otro lado el procesamiento GPU a través de CUDA. A continuación se presenta una tabla de comparación:

Tabla 7: Análisis comparativo de tiempo de procesamiento entre CPU y GPU

Tipo de Imagen	Tamaño de imagen	Tiempo de Procesamiento	
		CPU	GPU
Estándar de Televisión	640×480	$9{:}35\ m$	$7{:}47\ m$
	720×576	$14{:}28\ m$	$11{:}03\ m$
	720×478	$12{:}21\ m$	$9{:}00\ m$
	1280×720	$20{:}58\ m$	$15{:}12\ m$
Imágenes pequeñas	100×100	$21\ s$	$21\ s$
	300×300	$3{:}05\ m$	$2{:}52\ m$
	300×200	$1{:}55\ m$	$1{:}50\ m$
	455×569	$7{:}15\ m$	$6{:}55\ m$
	305×305	$2{:}17\ m$	$2{:}12\ m$
	201×201	$1{:}03\ m$	$0{:}57\ m$
	105×105	$0{:}16\ m$	$0{:}16\ m$

	$93KB$	$3{:}08\ m$	$3{:}01\ m$
	$119KB$	$9{:}54\ m$	$8{:}27\ m$

Fuente: Elaboración propia

En el anterior cuadro se demuestra que los algoritmos desarrollados y propuestos por distintos autores pueden ser mejorados con la aplicación del procesamiento en paralelo a través de CUDA y la posibilidad de utilizar objetos GPU en la manipulación de imágenes digitales. El cuadro además, permite observar que se ha cumplido el objetivo de mejorar los tiempos en el proceso del sistema criptográfico.

CAPÍTULO VII

CONCLUSIONES

7.1. CONSECUCION DE OBJETIVOS

Culminado el desarrollo y las pruebas realizadas del sistema criptográfico, se verifica si los resultados obtenidos representan el alcance de los objetivos propuestos al inicio del presente trabajo.

7.1.1. Consecución del objetivo principal

La solución desarrollada tiene como funcionalidad principal la encriptación de imágenes en escala de grises. Dicha solución fue testeada en base a tiempos de ejecución en la que se ha demostrado que el proceso resulta eficiente tomando en cuenta el tamaño de las imágenes utilizadas en las pruebas y evaluación.

Por otra parte, la robustez del algoritmo es demostrada en la aplicación del mapa logístico y su correspondiente ecuación que alcanza un comportamiento caótico. Dicho comportamiento genera un efecto importante en la intensión de minimizar cualquier riesgo de seguridad con la diferencia que no supone mayor tiempo de procesamiento. Un tiempo aceptable de procesamiento con la encriptación del caos garantiza la disponibilidad del recurso y confidencialidad en el contenido.

7.1.2. Consecución de objetivos específicos

De acuerdo a los objetivos específicos definidos al inicio, a continuación se explican la manera en los que fueron completados:

- El alto nivel de seguridad criptográfica fue alcanzado en base a la arquitectura presentada la encriptación dividida en dos etapas principales, confusión y difusión. En otras palabras, no solo se ha decido cambiar el valor de los pixeles, sino también desordenarlos de forma caótica pero controlable. La ejecución en seria de estas etapas permite mantener el control de la encriptación para que el proceso en reversa sea de igual manera en serie. Por otra parte, el número de iteraciones, a diferencia de otros algoritmos, es de acuerdo al tamaño de la imagen, de ahí el que el tiempo de proceso depende del tamaño de la imagen a cifrar.

- Aplicar herramientas de hardware externas para un procesamiento más rápido como CUDA, permitió mejorar los tiempos de ejecución en la encriptación como también en la desencriptación. Debido a que se tiene previsto manipular grandes cantidades de pixeles por cada imagen a cifrar, se vio conveniente aplicar el procesamiento en paralelo para que la tarjeta de video, en este caso NVIDIA, procese los pixeles en grupos de pixeles de forma paralela y no así de manera secuencial como lo ejecutan en otras investigaciones similares. El procesamiento en serie es llevado a cabo por la memoria CPU, mientas que el procesamiento en paralelo utiliza tecnología GPU para tal efecto. Por otra parte la versatilidad del lenguaje de programación utilizado, en este caso C++, fue el complemento ideal para manipular las librerías de código abierto de OpenCV que gestionan la memoria GPU.
- La decisión de aplicar conceptos de Teoría del Caos para encriptar imágenes fue a razón de una propiedad importante y singular referido al “Efecto mariposa”. Cualquier cambio mínimo en las claves o llaves secretas que sea utilizado por un atacante no genera una imagen semidescifrada aun cuando su valor sea muy cercano al valor correcto. De esta manera el sistema criptográfico se torna muy sensible ante ruidos externos gracias a la aplicación de una ecuación logística. Otra de las propiedades que otorga el comportamiento caótico es que genera números pseudo-aleatorios tanto en el intercambio de posiciones como en el cambio de valor de cada pixel lo cual hace mucho más improbable que el atacante pueda acertar en la semilla inicial de cada iteración. Lo que es más, la no-periodicidad de los números pseudo-aleatorios que genera el mapa logístico reduce de gran manera que el atacante pueda resolver la ecuación a través de algún elemento que se repita a lo largo de la serie generada. Por lo tanto, todas estas razones y propiedades que ofrece la teoría del caos eleva la robustez del sistema criptográfico desarrollado.

7.2. CONCLUSIONES

Para superar la problemática propuesta, este trabajo presenta un nuevo algoritmo de encriptación implementado en una arquitectura vanguardista y apoyada sobre nuevas tecnologías de procesamiento computacional. Actualmente, la seguridad para imágenes digitales se ha convertido altamente importante debido a que la comunicación por Internet a través de la transmisión de productos digitales ocurre con mayor frecuencia. Para tal motivo, la propuesta combina teoría del caos aplicados a un esquema de alta seguridad y repetitiva en forma variable. La constante de la ecuación logística y el valor inicial son utilizados como una llave de encriptación dentro del sistema caótico de encriptación. El análisis experimental demuestra que el algoritmo de encriptación de imágenes basado en el mapa logístico ofrece ciertas ventajas sobre el tamaño de llave y el nivel de seguridad, mientras mantiene y/o mejora los tiempos de procesamiento en comparación con otras investigaciones.

El presente sistema de criptografía es adecuado para la encriptación de imágenes que viajan a través de Internet y/o aplicaciones móviles que realizan el intercambio de este recurso multimedia. El algoritmo desarrollado combina las propiedades de la confusión y difusión como esquema principal del proceso de encriptación. Ambas etapas se desarrollan en base a mapas logísticos, una para cada una. Lo cual provee de mayor ergodicidad y amplitud en la clave secreta. Por otra parte, y no menos importante, la alta sensibilidad a ruidos externos provee alta seguridad e integridad a la imagen encriptada. En este caso, la sensibilidad se aplica sobre las condiciones iniciales de las ecuaciones logísticas. La diferencia del algoritmo propuesto en el presente trabajo con relación a otras investigaciones radica en la combinación del esquema de proceso en serie con la aplicación de mapa logístico y por otra parte el número de iteraciones variable de acuerdo a la cantidad de pixeles.

Aunque el presente algoritmo se enfoca en la encriptación de imágenes digitales, no está limitado a esta área y puede ser ampliamente aplicado en otros campos de seguridad de información.

7.3. LINEAS FUTURAS

Debido al crecimiento exponencial que tienen varias ramas de la tecnología en la que se incluye la seguridad, el presente trabajo propone las siguientes líneas futuras de investigación que pueden mejorar el rendimiento del algoritmo como ser:

- Aplicar mapas caóticos de dos y más dimensiones como por ejemplo el mapa de Baker, mapa de Arnold, entre otros. Este punto puede ser aún más estudiado para encontrar puntos comunes entre varios mapas caóticos y evaluar la combinación entre ellos durante el proceso de encriptación.
- Iterar un número determinado de veces por separado para confusión y difusión.
- Aplicar el algoritmo a imágenes a color donde el cálculo matemático debe ser realizado por separado para las tres matrices correspondientes a los colores principales RGB.
- Encriptar video cortos y de baja calidad en principio a través de los frames por segundo que se generen.

BIBLIOGRAFÍA

Alligood, K., Sauer, T., & Yorke, J. (2000). *CHAOS: An Introduction to Dynamical Systems.* New York: Springer-Verlag.

Belkhouche, F., & Qidwai , U. (2003). Binary image encoding using 1D chaotic maps. *IEEE Region 5, 2003 Annual Technical Conference*, 39 - 43.

Biblioteca de la Universidad de Cornell / Departamento de Investigación. (2003). *Tutorial de Digitalización de Imágenes - Gestión.* Obtenido de www.library.cornell.edu

Bradski, G., & Kaehler, A. (2008). *Learning OpenCV.* O'Reilly.

Charte , F. (2009). *Torre de Babel*. Obtenido de http://fcharte.com/Default.asp?noticias=2&a=2009&m=12&d=15

Chen, G., Mao, Y., & Chui, C. (2004). A symmetric image encryption scheme based on 3D chaotic. *Chaos, Solitons & Fractals*, 749–761.

Chen, L., & Zhao, D. (2008). Image encryption with fractional wavelet packet method. *Elsevier GmbH*, 286–291.

Cui, D. (2010). A Novel Fingerprint Encryption Algorithm Based on Chaotic System and Fractional Fourier Transform. *International Conference on Machine Vision and Human-machine Interface*, (págs. 168 - 171).

Delfs, H., & Knebl, H. (2006). *Introduction to Cryptography.* Nurnberg: Springer.

Ephin, M., Joy, J. A., & Vasanthi, N. A. (2013). Survey of Chaos based Image Encryption and Decryption Techniques. *International Journal of Computer Applications IJCA*, (págs. 1-5).

Gao, H., Zhang, Y., Liang, S., & Li, D. (2006). A new chaotic algorithm for image encryption. *Chaos, Solitons & Fractals*, 393–399.

Gao, T., & Chen, Z. (2008). Image encryption based on a new total shuffling algorithm. *Chaos, Solitons & Fractals*, 213 - 220.

Gobierno de España - Ministerio de Educación,Cultura y Deporte. (2015). *Instituto Nacional de Tecnologías Educativas y Formación del Profesorado.* Obtenido de http://www.ite.educacion.es/

Jiménez Rodríguez, M., Jaimes Reategui, R., & N. Pisarchik, A. (2012). Secure Communication Based on Chaotic Cipher and Chaos Synchronization. *Discontinuity, Nonlinearity and Complexity*, 57-68.

Lai, J., Liang, S., & Cui, D. (2010). A Novel Image Encryption Algorithm Based on Fractional Fourier. *International Conference on Multimedia Communications*, (págs. 24 - 27). China.

Lian, S., Sun, J., & Wang, Z. (2005). Security Analysis of A Chaos-based Image Encryption Algorithm. *Phisica A 351, Elsevier Science*, 645–661.

Lorenz, E. (2005). *The Esscence of Chaos.* London: University College London (UCL).

Mao, Y., Chen, G., & Lian, S. (2004). A novel fast image encryption scheme based on 3D chaotic baker maps. *International Journal of Bifurcation and Chaos*, 3613 - 3624.

Matthews, R. (1989). On the derivation of a chaotic encryption algorithm. *Cryptologia*, 29–42.

Moon, D., Chung, Y., Pan, S., Moon, K., & Chung, K. (2006). An Efficient Selective Encryption of Fingerprint Images for Embedded Processors. *ETRI Journal*, 444 - 452.

National Institute of Standards and Technology. (13 de Julio de 2009). *NIST*. Obtenido de http://museum.nist.gov/

NVIDIA Corporation. (2015). *NVIDIA*. Obtenido de http://www.nvidia.es/

NVIDIA CUDA. (2006). *NVIDIA GeForce 8800 GPU Architecture Overview.*

NVIDIA CUDA. (2007). *Compute Unified Device Architecture – Programming Guide.*

NVIDIA CUDA. (2008). *Installation and Verification on Microsoft Windows XP and Windows Vista.*

Ott, E. (2002). *Chaos in Dynamical Systems.* United Kingdom: Cambridge University Press.

Pareek, N., Patidar, V., & Sud, K. (2006). Image encryption using chaotic logistic map. *Image and Vision Computing*, 926–934.

Philip, M., & Das, A. (2011). Survey: Image Encryption using Chaotic Cryptography Schemes. *International Journal of Computer Applications IJCA Special Issue on "Computational Science - New Dimensions & Perspectives"*, (págs. 1 - 4).

Philip, M., & Das, A. (2011). Survey: Image Encryption using Chaotic Cryptography Schemes. *IJCA Special Issue on "Computational Science - New Dimensions & Perspectives"*, 1-4.

Pisarchik, A., & Zanin, M. (2008). Image encryption with chaotically coupled chaotic maps. *Elsevier B.V*, 2638–2648.

Rojas, Á. M., & Cano, A. R. (2011). Cifrado de imágenes y Matemáticas. *Revista Iberoamericana de Tecnología en Educación y Educación en Tecnología*, 30 - 37.

Sánchez, S. (2008). *Desarrollo de algoritmos de análisis de imágenes hiperespectrales en tarjetas gráficas NVidia.*

Sankpal, P. R., & Vijaya, P. A. (2014). Image Encryption Using Chaotic Maps: A Survey. *Fifth International Conference on Signals and Image Processing*, (págs. 102-107).

Sharma, M., & Kowar, M. K. (2010). IMAGE ENCRYPTION TECHNIQUES USING. *International Journal of Engineering Science and Technology*, 2359-2363.

Sprott, J. (2015). *Sprott's Gateway.* Obtenido de http://sprott.physics.wisc.edu/

Strien, S., & Melo, W. (2012). *One-Dimensional Dynamics.* Springer Science & Business Media.

Sun, F., Liu, S., Li, Z., & Lu, Z. (2008). A novel image encryption scheme based on spatial chaos map. *Chaos, Solitons & Fractals*, 631–640.

Thunberg, H. (2001). Periodicity versus Chaos in One-Dimensional Dynamics. *Society for Industrial and Applied Mathematics*, 3 - 30.

Weckesser, W. (2005). *One Dimensional Maps.* Department of Mathematics.

Zhang, L., Liao, X., & Wang, X. (2005). An image encryption approach based on chaotic maps. *Chaos, Solitons & Fractals*, 759–765.

Zhao, S., Li, H., & Yan, X. (2008). A secure and efficient fingerprint images encryption scheme. *The 9th International Conference for Young Computer Scientists* (págs. 2803 - 2808). IEEE Computer Society.